Naam:	
Nummer:	
Klas:	

Interactieve Media – Propedeuse – Projectwerk blok 3

In blok 2 en 3 werk je in een team aan een project – het bedenken en realiseren van een interactieve oplossing voor een probleem van een externe opdrachtgever.

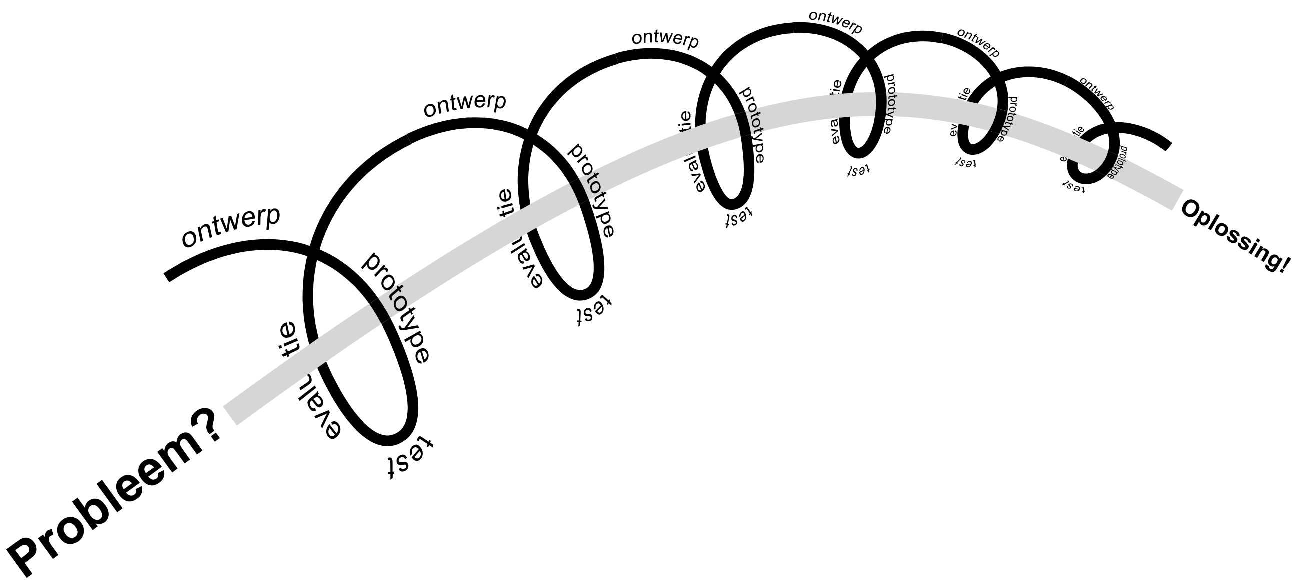
- Het project gaat in week 1 van blok 3 verder met de kick-further.
- In week 2 en 3 verifieer je de kwaliteit van je concept door de demo door gebruikers te laten testen.
- In week 10 presenteer je het product aan de opdrachtgever.

Deze manual is het projectspoorboekje voor blok 3. De manual bestaat uit drie delen:

Deel 1: Projectwerk

Toepassen van kennis, vaardigheden op een opdracht van een externe opdrachtgever

Het toepassen van (juist) verworven vaardigheden en kennis op reële opdrachten van externe opdrachtgevers is een belangrijk onderdeel van de opleiding. In de eerste drie weken heb je kennis gemaakt met projectwerk tijdens Game ON! In het tweede en derde blok van het jaar voer je als team een project uit voor een externe opdrachtgever. In het vierde blok voer je individueel een project uit – de afsluiting van het jaar. Projectwerk is de 'ruggengraat' binnen het jaar.



1.1 Wat is een project

In een organisatie worden allerhande werkzaamheden uitgevoerd. Deze kunnen grofweg in drie groepen verdeeld worden: 1. improvisatie, 2. routine en 3. projectwerk.

Nieuwe werkzaamheden en incidenten kunnen worden aangepakt door middel van improvisatie. Routinematige werkzaamheden worden herhaaldelijk uitgevoerd, zijn goed voorspelbaar, vaak in procedures beschreven en efficiënt.

Tussen routine en improvisatie ligt projectwerk. Projectwerk heeft een eenmalig en tijdelijk karakter en is redelijk voorspelbaar. Om de voorspelbaarheid te vergroten wordt planmatig gewerkt, zodat de te volgen weg geleidelijk duidelijk wordt. Dit betekent dat vooraf de tijd genomen wordt om na te denken over het te bereiken doel en de manier waarop dit doel bereikt kan worden. Het project wordt meestal opgesplitst in een aantal delen/fases. Voor projectwerk wordt meestal een aparte organisatie opgezet. Dit betekent dat mensen die normaal niet met elkaar werken, in een projectteam met een onderlinge taakverdeling samenwerken.

	Improvisatie	*Projectwerk*	*Routine*
Wanneer toepassen?	Bij ad hoc (plotseling) activiteiten	Bij voorziene activiteiten	Bij herhalend werk
Mate van voorspelbaarheid?	De uitkomst is onzeker	De uitkost is redelijk zeker	De uitkomst is zeker
Bekendheid van het werk?	Nieuw werk dat onverwacht boven komt	Nieuw werk dat gepland is	Bekend werk dat frequent uitgevoerd wordt
Mate van vrijheid?	Veel vrijheid bij uitvoering	Vooraf doordacht plan	Nauwelijks vrijheid
Werkwijze?	Chaotisch	Geleidelijk duidelijker	Duidelijk (beschreven in procedures)

Definitie: Een project is een tijdelijk werkverband van een aantal mensen – meestal uit verschillende vakgebieden – om een vooraf vastgesteld doel te bereiken met een vastgesteld budget voor een vastgestelde einddatum:

- Een project heeft een duidelijk ***startpunt*** – de kick-off.
- Een project is tijdelijk van aard – heeft een afgesproken ***einddatum***.
- Een project heeft een eenmalig en duidelijk gedefinieerd ***doel***. Het resultaat van een project is een ***eindproduct***: een website, een huis, een rapport, een gebeurtenis…
- Een project heeft een ***opdrachtgever*** die het project ‘betaalt’, die goedkeuring geeft en die indien nodig duidelijk verschaft en knopen doorhakt.
- Een project beschikt over een ***budget***: geld, middelen, mensen, tijd…
- In een project werken meestal mensen uit verschillende ***disciplines*** samen in een tijdelijk werkverband – het ***projectteam***.
- Een project heeft een eigen ***projectorganisatie***.
- In een project wordt planmatig gewerkt – volgens een ***plan van aanpak***.

Projectmatig werken is geen doel op zich. Wel is het een middel om werkzaamheden die minder voorspelbaar zijn structuur te geven en zo beter hanteerbaar te maken, opdat de kans op een succesvolle afloop zo groot mogelijk is.

bron: Projectmanagement – Roel Grit.

1.2 IAM Projectfasering

Binnen het vakgebied interactieve media wordt veel in projecten gewerkt. Om je optimaal op de beroepspraktijk voor te bereiden vormt projectwerk dan ook een belangrijk en terugkomend onderdeel van de opleiding. In het eerste jaar bedenk en realiseer je in blok 2 en 3 (in 18 weken) in een team een interactieve oplossing voor een probleem van een externe opdrachtgever.

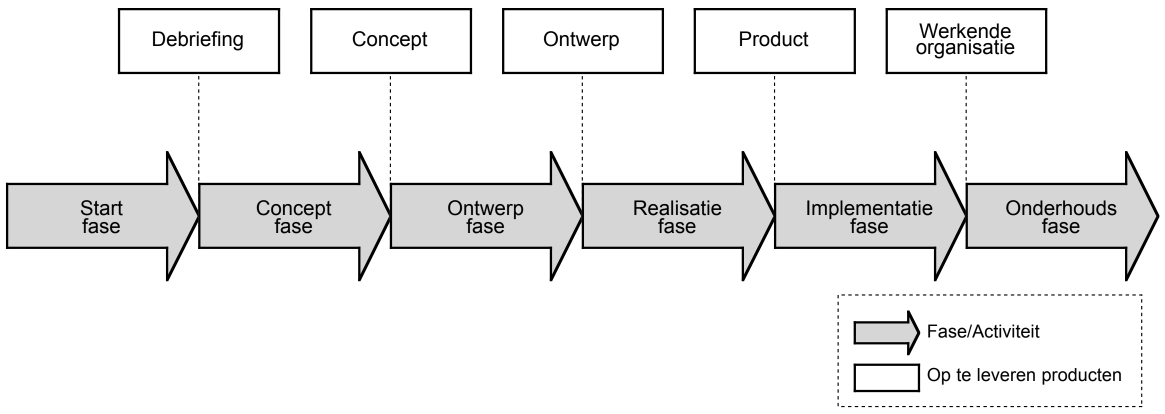
Projectwerk is aan de ene kant ***leuk*** en ***spannend*** omdat je met anderen op eigen wijze naar een onbekend eindresultaat toewerkt. Aan de andere kant is projectwerk ook ***lastig*** en soms ***frustrerend*** omdat teamleden het niet altijd met elkaar eens zijn, omdat de beschikbare tijd te kort is, omdat de opdrachtgever niet van een goed idee te overtuigen is, omdat content verzamelen meer tijd kost dan gedacht, omdat de techniek instabiel blijft, omdat teamleden afspraken niet nakomen, omdat de doelgroep het product afkeurt, omdat de concurrent met een beter product op de markt komt, omdat…

Planmatig werken

Om de kans op succes zo groot mogelijk te maken wordt een project planmatig aangepakt. Daartoe wordt het project in fases opgedeeld – aan het einde van elke fase wordt een deelproduct opgeleverd – een mijlpaal bereikt. Op deze manier wordt gedurende het project in de gaten gehouden of het team op de goede weg is, of dat aanpassingen in planning, op te leveren producten en/of benodigd budget nodig zijn. In het plan van aanpak worden alle afspraken vastgelegd: over het doel van het project, de planning, de fasering, de op te leveren producten, de inbreng van teamleden, de inbreng van de opdrachtgever, de inbreng van de doelgroep, de wijze van samenwerken…

Fasering en iteraties

In onderstaand diagram een algemeen gebruikte projectfasering binnen het vakgebied interactieve media. In de opeenvolgende fases wordt het idee steeds verder uitgewerkt tot uiteindelijk een werkend product wordt opgeleverd dat door de organisatie gebruikt wordt. Gedurende en aan het eind van elke fase wordt gecontroleerd of het werk voldoet aan de eisen: door het werk met de opdrachtgever te bespreken, maar vooral door het idee te laten testen door de doelgroep. Deze aanpak van herhaalde verfijning heet iteratief ontwikkelen: 1. idee bedenken, 2. uitwerken, 3. testen, 4. evalueren en weer 1. verfijnen, 2. uitwerken, 3. testen…



Startfase

In de startfase onderzoekt het team wat de opdrachtgever en de doelgroep nu eigenlijk precies willen – het projectdoel wordt vastgesteld. Het deelproduct dat aan het einde van de startfase wordt opgeleverd is de debriefing, waarin staat beschreven wat de opdrachtgever wil – de opdracht of probleemstelling – en hoe dit gerealiseerd gaat worden. Het team presenteert dit plan aan de opdrachtgever en andere belanghebbenden en start na goedkeuring met de volgende fase – de conceptfase.

Conceptfase
In de conceptfase bedenkt het team oplossingen voor het in het plan van aanpak beschreven probleem/doel van de opdrachtgever (en de doelgroep). Dit is *de cruciale fase* in het project, omdat uit vele mogelijkheden een definitieve richting gekozen wordt.

Het concept dat aan de opdrachtgever wordt gepresenteerd geeft allereerst een goed beeld van de kern van het idee en de functionaliteit. Daarnaast wordt het concept zo uitgewerkt dat de opdrachtgever zich een goed beeld kan vormen van het gehele idee: gebruikersscenario's, artwork, schermschets, tone of voice…

Ontwerpfase
In de ontwerpfase werkt het team het concept uit. Er worden één of meerdere (werkende) demos gemaakt om zo de interactieve aspecten van het ontwerp te kunnen beoordelen. Zeker in deze fase laat het team het product (of de demo) vaak één of meer maal testen door de doelgroep. Het ontwerp wordt uitgewerkt in specificaties die een compleet geven van het product: functioneel ontwerp, interactie ontwerp, visueel ontwerp, technisch ontwerp, copy concept en content matrix. Daarnaast wordt een verfijnde en eventueel aangepaste planning opgeleverd.

Realisatiefase
In de realisatiefase wordt het product gerealiseerd: de code wordt geschreven en getest, de database wordt ontworpen en ingericht, de benodigde visuals, geluiden en animaties worden gecreëerd, de content wordt verzameld, geredigeerd en getest. Het eindproduct van deze fase is een werkend product.

Hoe goed er ook in de concept en ontwerpfase is nagedacht, bij het realiseren van het product loop je als team altijd nog tegen een aantal problemen op, waarvoor oplossingen gevonden moeten worden. Ook tijdens de realisatie worden dus nog veel (kleine) beslissingen genomen. Indien er tijdens de realisatie echter nog grote hiaten naar boven komen, is het zaak om goed op te letten dat de kracht van het oorspronkelijke concept behouden blijft en om de oplossingen te bespreken met de opdrachtgever en eventueel opnieuw te laten testen door de doelgroep.

Naast het realiseren van het product wordt de implementatie van het product voorbereid.

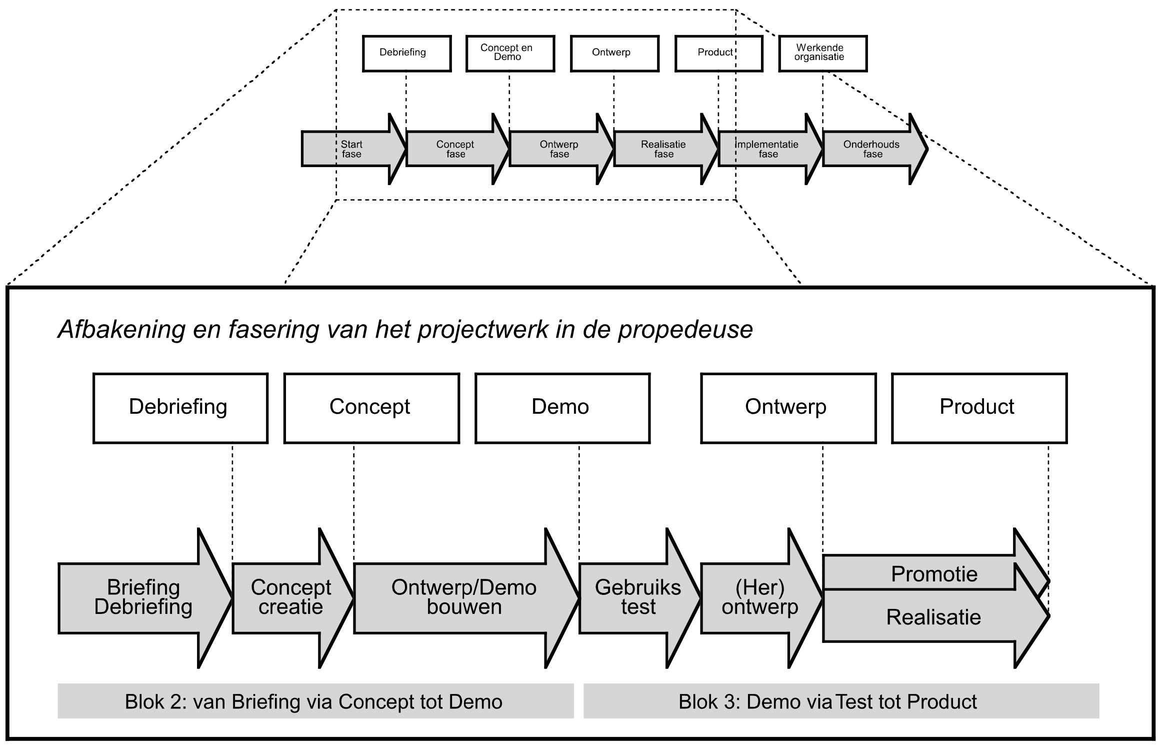
Implementatiefase
In deze fase draagt het team het product over aan de opdrachtgever. Het product wordt zowel technisch als organisatorisch ingepast in de organisatie. Het technische aspect is meestal niet het lastige onderdeel, al kan het inrichten van de server of het inrichten en koppelen van de database de nodige aandacht en voorbereiding vergen.

De organisatorische implementatie vraagt meestal meer aandacht. Een goede organisatorische inbedding is essentieel voor een succesvolle afronding van het project. Medewerkers moeten allereerst overtuigd worden van het nut van het nieuwe product (dit gebeurt bij voorkeur ook al eerder in het traject). Daarnaast moeten ze hun werkwijze aanpassen, getraind worden en bij het gebruik ondersteund worden. Het eindproduct van de fase is een werkend product dat naar tevredenheid gebruikt wordt.

Onderhoudsfase
Formeel stopt een project (meestal) na de implementatie van het product. Toch vindt het meeste werk vaak plaats na oplevering. Het beheren en up-to-date houden van content kost veel tijd. Dit zal meestal uitgevoerd worden door de medewerkers van de organisatie zelf, daarbij is vaak wel inhoudelijke ondersteuning nodig. Daarnaast komen bij het 'echte' gebruik van het product altijd toch nog functionele en technische problemen en wensen naar boven. Afhankelijk van de prioriteit worden die meteen opgelost of meegenomen naar een volgende versie. De onderhoudsfase is een voortdurend proces dat er op gericht is om de continue werking en kwaliteit van het product na oplevering te waarborgen en te verbeteren.

1.3 Speerpunten propedeuseproject

Gedurende de opleiding komen door de jaren heen in verschillende projecten alle facetten van projectwerk en onderhoud aan bod. Voor het propedeuseproject is een aantal speerpunten gedefinieerd, zoals weergeven in onderstaand diagram:



Uitgangspunten

- Studenten doorlopen het ***gehele ontwikkeltraject*** van start tot oplevering.
- De nadruk ligt op het traject **van *conceptontwikkeling tot realisatie***:
  - de studenten starten met een helder gedefinieerde probleemstelling,
  - het project wordt afgesloten met de presentatie van het product.
- Studenten werken op ***iteratieve wijze*** naar het eindproduct toe.
- Studenten werken op ***planmatige wijze*** en leveren geregeld deelproducten op.
- Het ontwerpen vóór en verifiëren van de kwaliteit van ideeën en ontwerpen dóór de ***doelgroep*** is een natuurlijk en essentieel onderdeel van het projectwerk.
- Studenten werken in een ***projectteam*** aan het bedenken en realiseren van een interactieve oplossing voor een probleem van een ***externe opdrachtgever***.
- Het probleem van de opdrachtgever biedt ruimte voor ***creatieve oplossingen***:
  - studenten worden uitgedaagd buiten gebaande paden te denken.
- De studenten worden ondersteund met een ***fasering en planning op hoofdlijnen***.

Doelstelling/Succesindicatoren

- Studenten zijn ***trots op hun werk***.
- Studenten zijn ***trots op hun team***.
- Alle teams hebben een ***werkend product*** opgeleverd.
- De ***producten zijn bijzonder, divers en hebben een 'interactief' karakter***.
  - Studenten kiezen niet voor een voor de hand liggende oplossing.
- Studenten kunnen na afloop de ***fases en kritische momenten benoemen***.
- Opgedane ervaring, kennis en vaardigheden bieden ***een opmaat voor projecten met grotere complexiteit***.
- Opgedane ervaring, kennis en vaardigheden bieden ***een opmaat voor verdere professionalisering***.

1.4 Teams en Begeleiding

Binnen het projectwerk wordt intensief door de studententeams, de opdrachtgever en de teamcoach samengewerkt – elk met hun eigen rol en verantwoordelijkheid.

Studententeams

Teams worden samengesteld uit 4 tot 6 studenten uit één klas (5 teams per klas).
Twee klassen werken aan hetzelfde project (10 teams per opdracht).
Elk team stelt één contactpersoon aan, die contact onderhoudt met de opdrachtgever.
Elk team stelt één contactpersoon aan, die contact onderhoudt met de teamcoach.

Opdracht en Opdrachtgever

De opdracht is een reëel probleem van een externe opdrachtgever – de opdrachtgever. Het project eindigt aan het eind van het derde blok met de presentatie van alle producten en de implementatie en ingebruikname van de levensvatbare producten. Mocht het niet haalbaar zijn om een product in het derde blok te implementeren, dan kunnen na afloop van het project afspraken gemaakt worden tussen het team, de opdrachtgever en de teamcoach omdat alsnog te doen in het vierde blok. In het curriculum is in het vierde blok ruimte om die extra activiteiten te belonen met studiepunten.

De rol van de opdrachtgever bestaat uit de volgende activiteiten:

- Het briefen van de teams bij de aftrap van het project.
- Het beantwoorden van vragen van teams gedurende het project.
- Het verzamelen en aanleveren van content (na een redelijke termijn).
- Het ondersteunen van de teams bij het uitvoeren van de gebruikstest.
- Het ondersteunen van de teams bij de (eventuele) implementatie.
- Het bekijken van, feedback geven op en accepteren van de deelproducten, het eindproduct en presentaties.

Teamcoach en Begeleiding

Elk studententeam heeft een teamcoach, waarmee wekelijks het werk besproken wordt. In deze teamcoach-gesprekken worden de inhoud, de voortgang, de planning, het teamwerk en vragen en problemen besproken en worden presentaties voorbereid. Incidenteel zullen teamcoaches ook de voortgang van elkaars teams bespreken, opdat het werk vanuit verschillende invalshoeken en expertises bekeken wordt.

Daarnaast worden de studenten ondersteund met een serie workshops en colleges waarin algemene projectvaardigheden besproken en geoefend worden. Deze workshops en colleges worden verzorgd door de teamcoaches, en indien relevant door andere docenten van de opleiding en/of gastdocenten/sprekers.

De rol van de teamcoach bestaat uit de volgende activiteiten:

- Het wekelijks bespreken van de voortgang met de projectteams.
- Het verzorgen van ondersteunende workshops en colleges.
- Het ondersteunen van de teams bij het opzetten en uitvoeren van de gebruikstest.
- Het ondersteunen van de teams met documenteren en bij de presentaties en de (eventuele) implementatie.
- Het bekijken en beoordelen van de deelproducten, het eindproduct en presentaties.
- Het na afloop van elk blok evalueren van het projectwerk met de projectteams.
- Het bemiddelen bij conflicten binnen projectteams.

Manual

Elk blok ontvangt elke student een project manual. In deze manual staat informatie over de grote lijn van het project, het programma, de planning, de op te leveren producten en de wijze van beoordelen.

1.5 Beoordeling: product, proces en presentatie

Voor het project zijn in totaal 10 studiepunten te verdienen (4 in blok 2 en 6 in blok 3). Bij de beoordeling van het werk van de teams en individuele studenten wordt gekeken naar de volgende aspecten:

- ***product*** (inhoud van de verschillende deelproducten en het eindresultaat)
- ***proces*** (het ontwerpproces en het teamwerk)
- ***presentatie*** (communicatie met teamcoach en opdrachtgever, schriftelijke rapportage en (mondelinge) presentaties)

Aan het eind van elk blok wordt het werk beoordeeld. De teamcoach is verantwoordelijk voor de beoordeling. De opdrachtgever geeft de teamcoach advies over de kwaliteit van de opgeleverde producten en de wijze waarop het werk gepresenteerd wordt.

De beoordeling wordt besproken in een evaluatie na afloop van het blok. Tijdens deze evaluatie beoordelen de studenten elkaars prestaties onderling – de teambeoordeling.

Van elk team wordt een projectdossier bijgehouden, dat gebruikt wordt bij het beoordelen van het proces. In dit dossier worden de volgende onderdelen opgenomen:

- opgeleverde deelproducten en verslagen
- hand-outs van presentaties
- verslagen van de teamcoach-gesprekken
- feedback van opdrachtgever
- procesverslagen van teamleden
- teambeoordelingen
- beoordelingen van teamcoaches
- afspraken met teamcoaches
- waarschuwingen van teamcoaches

Teambeoordeling als basis

Na blok 2 kunnen de volgende teambeoordelingen gegeven worden:

- ***Voldoende tot Uitmuntend***: Het team is op de goede weg om het gehele project met minstens een voldoende beoordeling af te ronden.
- ***Voldoende maar***: Het werk, het proces en/of de presentatie van het team zijn net aan van voldoende niveau. Als het team op dezelfde wijze doorgaat zal blok 3 met een onvoldoende beoordeling worden afgesloten. In de evaluatie na afloop van blok 2 worden concrete afspraken gemaakt tussen de teamcoach en het team over wat het team in blok 3 moet presteren om ook blok 3 minimaal met voldoende resultaat af te ronden.
- ***Onvoldoende***: Het werk, het proces en/of de presentatie zijn niet van voldoende niveau. In de evaluatie na blok 2 worden concrete afspraken gemaakt tussen de teamcoach en het team over wat het team in blok 3 moet presteren om het gehele project met voldoende resultaat af te sluiten. Komt het team de afspraken na, dan wordt het projectwerk in blok 2 alsnog voldoende beoordeeld.

Na blok 3 kunnen de volgende teambeoordelingen gegeven worden:

- ***Voldoende tot Uitmuntend***: Het team heeft het project met succes afgerond.
- ***Onvoldoende maar***: Het werk, het proces en/of de presentaties zijn niet van voldoende niveau maar bieden genoeg aanknopingspunten om met behulp van een aanvulling in de eerste weken van blok 4 alsnog een voldoende te behalen. In de evaluatie na blok 3 worden concrete afspraken gemaakt tussen de teamcoach en het team aan welke voorwaarde de aanvulling moet voldoen en wanneer deze gereed is.
- ***Onvoldoende***: Het werk, het proces en/of de presentaties zijn niet van voldoende niveau en geven geen aanleiding om het werk aan te vullen.

Individuele aanpassing

De beoordeling van het team vormt de basis voor de beoordeling van de individuele student. Bij bovenmatige of ondermaatse prestaties van een student kan de beoordeling naar boven dan wel naar beneden bijgesteld worden. De teamcoach is verantwoordelijk voor deze individuele aanpassingen. De conclusies uit de teamevaluatie – waarin de studenten elkaars prestaties beoordelen – zijn hierbij adviserend.

Na blok 2 kunnen de volgende individuele beoordelingen gegeven worden:

- ***Voldoende tot Uitmuntend***: De student is op de goede weg om het gehele project met minstens een voldoende beoordeling af te ronden.
- ***Voldoende maar***: De inzet, het gedrag en/of de inhoudelijke bijdrage van de student zijn net aan van voldoende niveau. Als de student op dezelfde wijze doorgaat zal blok 3 met een onvoldoende beoordeling worden afgesloten. In de evaluatie na blok 2 worden concrete afspraken gemaakt tussen de teamcoach en de student over wat de student in blok 3 moet presteren om het projectwerk ook blok 3 minimaal met voldoende resultaat af te ronden.
- ***Onvoldoende***: De inzet, het gedrag en/of de inhoudelijke bijdrage van de individuele student zijn van onvoldoende niveau. Een individuele bijstelling naar beneden mag alleen tot een onvoldoende beoordeling leiden als de student in de loop van het blok minimaal één maal gewaarschuwd is door de opleiding (deze waarschuwing is vastgelegd in het teamdossier) of als de student niet heeft deelgenomen aan het projectwerk.
  In de evaluatie na blok 2 worden concrete afspraken gemaakt tussen de teamcoach en de student over wat de student in blok 3 moet presteren om het gehele project met voldoende resultaat af te ronden. Komt de student de afspraken na, dan wordt het werk in blok 2 alsnog voldoende beoordeeld.

Na blok 3 kunnen de volgende individuele beoordelingen gegeven worden:

- ***Voldoende tot Uitmuntend***: De student heeft het project met succes afgerond.
- ***Onvoldoende maar***: De inzet, het gedrag en/of de inhoudelijke bijdrage van de student zijn niet van voldoende niveau maar bieden genoeg aanknopingspunten om met behulp van een individuele aanvulling in de eerste weken van blok 4 alsnog een voldoende te behalen. In de evaluatie na blok 3 worden concrete afspraken gemaakt tussen de teamcoach en de student aan welke voorwaarde de aanvulling moet voldoen en wanneer deze gereed is.
- ***Onvoldoende***: De inzet, het gedrag en/of de inhoudelijke bijdrage van de student zijn niet van voldoende niveau en geven geen aanleiding om het werk aan te vullen. Een individuele bijstelling naar beneden mag alleen tot een onvoldoende beoordeling leiden als de student in de loop van het blok door de opleiding minimaal één maal gewaarschuwd is (deze waarschuwing is vastgelegd in het teamdossier) of als de student niet heeft deelgenomen aan het projectwerk.

Individuele uitsluiting

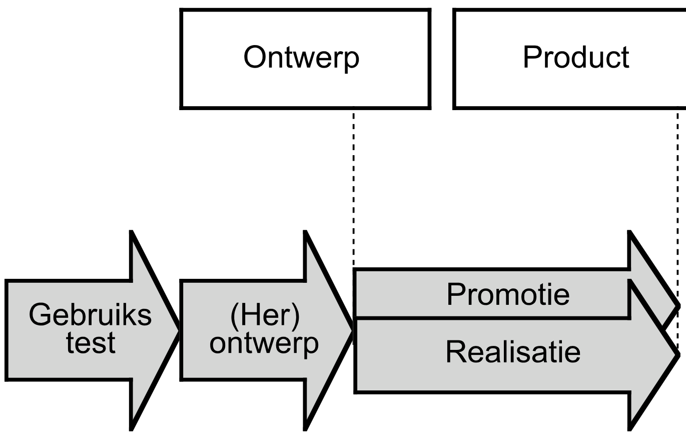
Bij voortdurend ondermaats presteren of onaangepast gedrag van een individuele student kan de student uit het team gezet worden en verdere deelname aan het projectwerk ontzegd worden. De teamcoach is verantwoordelijk voor deze ontzegging. Voordat een student wordt uitgesloten van verdere deelname, moet deze door de opleiding minimaal twee maal gewaarschuwd zijn. Deze waarschuwing inclusief afspraken is vastgelegd in het teamdossier. Daarnaast moet de student de kans hebben gehad om zijn prestaties, inzet of houding te verbeteren.

Deel 2: Programma

Van Demo met Test tot Product

Het projectwerk in het propedeusejaar is verdeeld over 2 blokken. Blok 2 start met de kick-off en wordt afgesloten met het presenteren van een demo. Blok 3 start met de kick-further en het testen van de demo en wordt afgesloten met het presenteren van het product:

Week 1:	Kick-further
Week 2 en 3:	‘Usability’ test
Week 4:	Opleveren UI testrapport (bijlage van Ontwerp specificatie)
	Opleveren Ontwerpspecificatie resulterend in Go/NoGo
	Opleveren Planning
Week 10:	Opleveren Documentatie en Product 1.0
	Product presentatie voor opdrachtgever
	Beoordeling bespreken en evaluatie van het projectwerk



2.1 Programmaoverzicht

Het programma bestaat uit ondersteunende colleges op maandag en overleg met de teamcoach en workshops op donderdag. Het blok is onderverdeeld in vier fases: 1a. UI test, 1b. (Her)ontwerp, 2. Realisatie en 3. Presentatie. Bijzondere activiteiten, oplevermomenten en presentatiemomenten zijn grijs gemarkeerd.

Lesweek		**Colleges** *(maandag)*	**Teamcoach/Workshop** *(donderdag)*
Lesweek 1 kalender 5	*Evaluatie/UI test/ (Her)Ontwerp*	**Kick-further** *28 januari* - Kick-further - Usabilitytests	**Teamcoach** *31 januari* - Teambeoordeling - Beoordeling / Evaluatie blok 2
			Training *31 januari* - Instructie UI testlab (één student per team)
Lesweek 2 kalender 6		**College** *4 februari* - Architectuur	
		Testlab *op afspraak*	**Testlab** *op afspraak*
			Teamcoach *7 februari* - Bespreken UI test / Ontwerp
Lesweek 3 kalender 7		**College** *11 februari* - Specificaties & Content	
		Testlab *op afspraak*	**Testlab** *op afspraak*
			Teamcoach *14 februari* - Bespreken UI test / Ontwerp
Lesweek 4 kalender 8		**College** *18 februari* - Projectmanagement	**Teamcoach** *21 februari (workshop)* - Plannen
			Opleveren *22 februari (vrijdag)* - Ontwerpspecificatie - Planning (voor 13.00 uur)
Lesweek 5 kalender 9			
Lesweek 6 kalender 10	*Realisatie*	**College** *3 maart* - Testen	**Teamcoach** *6 maart* - Feedback Ontwerpspecificatie - Bespreken Voortgang / Planning
Lesweek 7 kalender 11		**College** *10 maart* - Implementatie	**Teamcoach** *13 maart* - Bespreken Voortgang / Planning
Lesweek 8 kalender 12		**College** *17 maart* - Promotie + Presentatie	**Teamcoach** *20 maart* - Bespreken Voortgang / Planning
Lesweek 9 kalender 13			**Teamcoach** *27 maart* - Bespreken Voortgang / Planning - Bespreken Presentatie
Lesweek 10 kalender 14	*Presentatie/Evaluatie*		**Opleveren** *2 april (woensdag)* - Product 1.0 - Documentatie (voor 13.00 uur)
			Product presentatie *3 april (donderdag)* - Presentatie van het product (volgens afspraak) - Voorselectie Golden Dot Awards
			Evaluatie *4 april (vrijdag)* - Teambeoordeling - Beoordeling / Evaluatie blok 3

Vak:	**Projectwerk – team (PPWT2, 3)**
Docenten:	Marjolijn Ruyg, Charlie Mulholland, Justus Sturkenboom, Theo Ploeg, Robbert Ritmeester, Frank Kloos
Periode:	(blok 2 en) blok 3
Aantal lesweken:	10 lesweken (inclusief feedback)
Aantal studiepunten:	(4 studiepunten en) 6 studiepunten in blok 3

Leerdoelen (blok 2 en 3)

- De student kan in een projectteam werken.
- De student kan de kern van een projectopdracht definiëren.
- De student kan een interactieve, oorspronkelijke en op de doelgroep afgestemde oplossing voor een probleem van een opdrachtgever bedenken en realiseren, waarbij de kracht van het initiële concept behouden blijft.
- De student kan keuzes maken en deze onderbouwen.
- De student kent het belang van planmatig werken – kan een actie- en besluitenlijst opstellen en deze effectief benutten.
- De student is bekend met de projectfases:
  start→ concept→ ontwerp→ UI test→ (her)ontwerp→ realisatie
- De student kent de rol van de volgende projectactoren:
  opdrachtgever, opdrachtnemer en gebruiker.
- De student kan een idee ‘schriftelijk’ en ‘mondeling’ overtuigend presenteren.
- De student kan een product sprankelend en passend presenteren.
- De student heeft oog voor detail en toont liefde voor het ambacht.

Plaats in het leerplan

Projecten nemen een speciale plaats in binnen het curriculum. In de projecten passen studenten al hun (juist verworven) kennis en vaardigheden toe op een opdracht van een externe opdrachtgever. De projecten zijn de plaats binnen de propedeuse waar studenten aantonen dat ze hun kennis, vaardigheden en houding samenhangend en effectief kunnen toepassen in beroepssituaties – de plaats waar studenten competentiegericht werken. Door het behalen van de projecten toont de student aan de eindcompetenties van de opleiding op het eerste niveau als geheel te beheersen.

Werkwijze

Studenten bedenken en creëren met een team een oplossing voor een probleem van een externe opdrachtgever. Aan het einde van blok 2 wordt een demo van de oplossing gepresenteerd. Aan het einde van blok 3 wordt het product gepresenteerd. Het project start met een kick-off – kennismaking met de opdracht, het team, de opdrachtgever en de teamcoach. Het programma bestaat uit een reeks werkgroepen en colleges waarin de studenten kennis maken en oefenen met algemene projectbegrippen, kennis en vaardigheden. Daarnaast bespreken de projectteams de voortgang wekelijks met de teamcoach en presenteren zij de resultaten een aantal maal aan de opdrachtgever, de teamcoach en andere studententeams.

Lesstof

- Teamwerk, projectmatig werken, methodisch werken
- Analyse, personae, scenario’s
- Creativiteit, schetsen, ideeën, concepten, selectie
- Presenteren, rapporteren en specificeren
- Iteratief werken: ontwerp, demo, test, herontwerp
- Bouwen, testen, implementeren

Criteria
De gedetailleerde eisen die aan de verschillende deelproducten (ontwerpspecificatie, planning, product, documentatie en presentatie) gesteld worden, staan op de volgende pagina's beschreven.

Product	1. Het gerealiseerde product werkt en is een interactieve, oorspronkelijke en op de doelgroep afgestemde oplossing voor het probleem van de opdrachtgever.
Presentatie	2. De documentatie geeft een compleet overzicht van het product en biedt de organisatie van de opdrachtgever voldoende houvast om het product na implementatie te gebruiken en te beheren. 3. De presentaties van het product boeien en overtuigen. 4. De opgeleverde producten en presentaties zijn verzorgd uitgevoerd en zijn geschreven in correct Nederlands. 5. De opgeleverde producten en presentaties vormen een bij het team en bij de opdracht passende eenheid.
Proces	6. De opeenvolging van deelproducten geven een goed beeld van de aanpak en de keuzes van het team 7. De student heeft actief deelgenomen aan het lesprogramma. 8. De student kan in de teambeoordeling zowel de eigen prestaties en werkwijze als die van de andere teamleden en het gehele projectteam op reële wijze evalueren.

Toetsing
In de loop van blok 3 worden de volgende producten opgeleverd:
– Lesweek 4: Opleveren van ontwerpspecificatie en planning
– Lesweek 10: Opleveren product en documentatie / Presenteren product

Beoordeling en feedback:
– Na oplevering van een deelproduct ontvangt het team feedback van zowel de opdrachtgever als de teamcoach.
– Na de oplevering van het product 1.0 wordt de beoordeling van het team bepaald door de teamcoach door te toetsen aan de criteria.
– In de teambeoordeling na afloop van het blok wordt de beoordeling voor individuele studenten naar boven of beneden bijgesteld door de teamcoach indien de uitkomsten van de teambeoordeling daartoe aanleiding geven.

Voor een voldoende beoordeling:
- Is aanwezigheid bij de kick-further verplicht.
- Moeten alle deelproducten tijdig opgeleverd worden.
- Moet de grond van eventuele onvoldoende beoordelingen van deelopdrachten of blok 2 aangepakt zijn bij de presentatie van het product.
- Moet het product gepresenteerd worden.
- Is aanwezigheid bij de presentaties verplicht.

Herkansing
Mocht een team of een individuele student een onvoldoende beoordeling ontvangen, dan maakt het team dan wel de student afspraken met de teamcoach over wat het team dan wel de student in blok 4 moet presteren om het gehele project met voldoende resultaat af te sluiten. Komt het team dan wel de student de afspraken na, dan wordt het projectwerk in blok 3 alsnog voldoende beoordeeld.

2.2 UI test

Het uitvoeren van een UI test (usability/user interface test) geeft inzicht in de sterke en zwakke punten van je ontwerp. De inbreng van eindgebruikers is hierbij onontbeerlijk hoe goed je zelf ook nadenkt over je ontwerp of hoe enthousiast de opdrachtgever ook is. De resultaten uit een UI test gebruik je om de zwakke punten in je ontwerp aan te passen maar vooral ook om de sterke punten te behouden en te versterken.

In lesweek 2 en 3 voert elk team een UI test uit. Hiervoor is een testlab ingericht waar proefpersonen met de ontwikkelde demo kunnen werken. Hierbij worden zowel de handelingen die de proefpersonen uitvoeren als de gezichtsuitdrukking van de proefpersonen opgenomen. Terwijl de proefpersonen met de demo werken kan het team het verloop van de tests in een andere ruimte volgen en bevindingen vastleggen. Na afloop worden de opnames en bevindingen bekeken en besproken met de proefpersonen. In het verdere verloop van het project kunnen teams het lab opnieuw reserveren om verbeteringen aan het product nogmaals te testen.

Voorbereiding

Het voorbereiden van een 'usability' test kost op zich niet veel tijd. De doorlooptijd is daarentegen meestal wel groot – start dan ook tijdig met voorbereiden:

- Maak met je teamcoach een afspraak wanneer in week 2 of 3 je de 'usability' test gaat uitvoeren.
- Zoek **drie proefpersonen** uit de doelgroep. Dit is vaak lastig – start hier tijdig mee.
- Stel een testplan op met daarin:
  - De testdoelen – waar ga je in het bijzonder op letten
  - De opdrachten/scenario's voor de proefpersonen
  - De taakverdeling – welk teamlid instrueert en begeleid de proefpersonen, welke teamleden observeren
- Verfijn **de demo** zodat zeker de door de proefpersonen te volgen scenario's goed uitgewerkt zijn. Zorg dat niet geïmplementeerde functionaliteit elegant is afgeschermd zodat technische storingen de test niet onderbreken.
- Zorg dat je de demo zelf goed test.
- Maak een schriftelijke opdracht/instructie voor de drie proefpersonen.
- Maak observatieformulieren waarop tijdens de test gemakkelijk aantekeningen gemaakt kunnen worden.

Uitvoering

- Zorg dat je op tijd bent.
- Heet de proefpersoon welkom en stel ze op hun gemak.
- Geef een korte en eenvoudige instructie over doel, inhoud en duur van de test. Overhandig de opdracht.
- Houd je mond tijdens de test – geef geen hints of aanvullende instructies.
- Vraag proefpersonen hardop te denken – moedig dit ook tijdens de test aan.
- De observanten letten uiteraard op de handelingen van de proefpersonen (zowel positieve als negatieve punten) – maar letten daarnaast ook op twijfels, terloops ter sprake gebrachte verbetervoorstellen, op jargon, gelach, op gezichtsuitdrukking, op andere non-verbale communicatie...

Verwerking

Direct na de tests worden de bevindingen en opnames met de proefpersonen door-gesproken. Let er hierbij op dat je geen suggestieve vragen stelt, maar zoekt naar verduidelijking van het gedrag van de proefpersonen dat je hebt waargenomen. Elk team krijgt de opnames met annotaties van de tests na afloop ter beschikking. Deze kunnen samen met de bevindingen gebruikt worden om verbetervoorstellen te doen. Doe dit zo snel mogelijk na de tests – de ervaring leert dat waarnemingen snel vertroebelen (zelfs als deze op video worden vastgelegd). De opnames kunnen daarnaast gebruikt worden om de opdrachtgever te overtuigen van de sterke punten van of verbetervoorstellen voor het product.

In het testlab wordt gebruik gemaakt van het softwarepakket The Observer XT (van Noldus). Met behulp van The Observer XT kunnen de opnames dan nogmaals bekeken en geanalyseerd worden. Met de software kan ook makkelijk een clip gemaakt met relevante fragmenten om bijvoorbeeld aan de opdrachtgever te tonen. Op 31 januari wordt een workshop aangeboden, waarin uitgelegd wordt hoe de software van Observer XT werkt. Aan deze workshop doet één teamlid uit elk team mee.

Op te leveren producten	*Voor*	*Datum/Locatie*
1. UI testrapport	Teamcoach	week 4 – 22/2 2008 (voor 13.00 uur)
Nb. Het testrapport wordt als bijlage opgenomen in de ontwerpspecificatie voor de teamcoach. De teams krijgen in het teamcoach-gesprek in week 6 feedback op het UI testrapport.		

1. UI testrapport
Het UI testrapport bevat de volgende onderdelen:
- ***Testaanpak***: De testdoelen, de opdrachten/scenario's, de taakverdeling en de proefpersonen
- ***Bevindingen***: De belangrijkste positieve en negatieve bevindingen
- ***Uiwerkingen:*** Verloop van de test van elke proefpersoon.

Randvoorwaarden
- Het testrapport wordt als bijlage opgenomen in de ontwerpspecificatie voor de teamcoach
- 2 A4 (exclusief uitwerkingen)
- Verzorgd uitgevoerd en passend in de teamstijl
- Passende presentatie van informatie in tekst, diagrammen en/of beeld
- Geschreven in correct Nederlands

2.3 (Her)ontwerp

In de ontwerpfase worden de details van het product ingevuld. Het ontwerp wordt uitgewerkt in specificaties die een volledig beeld geven van alle aspecten van het product. Hierbij wordt een evenwicht tussen compleetheid en efficiëntie nagestreefd, zodat het aantal verrassingen in de realisatiefase zo klein mogelijk blijft zonder dat het specificeren te veel tijd kost en een bureaucratisch proces wordt. Op het moment dat het ontwerp gedetailleerd is uitgewerkt, kunnen de planning en de taakverdeling aangepast en gedetailleerd worden.

Op te leveren producten	***Voor***	***Datum/Locatie***
1. Ontwerpspecificatie	Opdrachtgever	week 4 – 22/2 2008 (voor 13.00 uur)
2. Ontwerpspecificatie (uitgebreid)	Teamcoach	week 4 – 22/2 2008 (voor 13.00 uur)
3. Planning	Teamcoach	week 4 – 22/2 2008 (voor 13.00 uur)
Nb. De teams krijgen in het teamcoach-gesprek in week 6 feedback op de ontwerpspecificatie en de planning. De planning is de basis voor het verdere verloop van het projectwerk van het team en wordt elke week met de teamcoach doorgenomen en indien nodig aangepast.		

1. Ontwerpspecificatie

Het compleet uitwerken en specificeren van het ontwerp heeft vele functies:

- Geeft de opdrachtgever de informatie om te besluiten om al dan niet te starten met de qua tijdbesteding meestal dure realisatiefase – Go/NoGo.
- Geeft het team de informatie om een planning op te stellen.
- Geeft het team de informatie om een productieplan/content matrix op te stellen.
- Geeft de verschillende teamleden (rollen) de informatie om samen effectief aan één product te kunnen werken.
- Geeft het team de informatie om een testplan op te stellen.
- Vormt de basis voor de documentatie en handleidingen.

De ontwerpspecificatie voor de opdrachtgever is een aangepaste en completere versie van de demo specificatie en bevat de volgende onderdelen:

- ***What's new:*** Opsomming van de aanpassingen en aanvullingen ten opzichte van de demo specificatie
- ***Kernidee***: Omschrijving van de kracht van het idee in één zin
- ***Functionaliteit***: Omschrijving van de functionaliteit (per doelgroep) inclusief prioritering (volgens MoSCoW – Must, Should, Could en Won't have this time)
- ***Interactie***: Uitgewerkt als geheel in schermverloopdiagrammen of click-flows en in detail in specificaties van de afzonderlijke schermen/statussen (inclusief uitzonderingen en foutafhandeling)
- ***Grafisch***: Uitgewerkt in een stijlgids (kleur, stijl, typografie...) en in voorbeelden
- ***Copy***: Uitgewerkt in thesaurus (woordenlijst) en copy concept (tone of voice)
- ***Technisch***: Uitgewerkt in data model en technische architectuur
- ***Content matrix***: Opgenomen als bijlage (zie onderstaand)

2. Ontwerp specificatie (uitgebreid)

De ontwerpspecificatie voor de teamcoach bestaat uit de ontwerpspecificatie voor de opdrachtgever met als bijlage aangevuld met het ***UI testrapport*** (zie paragraaf 2.2)

Randvoorwaarden

- 10 à 15 A4 (inclusief kaft, exclusief bijlagen)
- Verzorgd uitgevoerd en passend in de teamstijl
- Passende presentatie van informatie in tekst, diagrammen en/of beeld
- Geschreven in correct Nederlands
- Bevat teamnaam, klas en namen en studentnummers van teamleden

Ad. Content matrix

Aan de ene kant geldt ‘Content is King’. Aan de andere kant is content verzamelen, structureren en redigeren een moeilijke en tijdrovende activiteit. Zeker als content moet worden aangeleverd door de opdrachtgever of andere externe partijen is de doorlooptijd daarbij vaak groot en wordt een goed contentplan onmisbaar. Een content matrix is samen met de planning, en de ontwerp specificatie een hulpmiddel om dit proces te structureren.

Een content matrix bevat de volgende onderdelen (de omvang is afhankelijk van het ontworpen product):

- ***Gedetailleerd overzicht van benodigde content***
- Tekst, beeld, geluid, animatie, video...
- Bron/Leverancier
- Benodigde middelen/Bewerkingen
- Naamgeving/Type/Bestandsformaat
- Planning/Deadline

3. Planning

De planning bestaat uit een actielijst en een besluitenlijst. De actielijst en de besluitenlijst zijn levende documenten – m.a.w. de lijsten worden voortdurend bijgewerkt en wekelijks doorgenomen met de teamcoach.

De actielijst bevat de volgende onderdelen:

Nr	*Onderwerp*	*Eigenaar*	*Status*	*Datum gepland*	*Datum gereed*	*Uren begroot*	*Uren gerealiseerd*	*Prioriteit*

De besluitenlijst bevat de volgende onderdelen:

Nr.	*Omschrijving*	*Datum*

2.4 Realisatie/Specificatie/Presentatie
In de realisatiefase wordt het product gerealiseerd: de code wordt geschreven en (technisch) getest, de database wordt ontworpen en ingericht, de benodigde visuals, geluiden en animaties worden gecreëerd, teksten worden verzameld, geredigeerd en getest en alle onderdelen worden samengevoegd. Het eindproduct van deze fase is een werkend product.

Naast het realiseren van het product wordt in deze fase de documentatie bijgewerkt en geschreven en wordt de presentatie voorbereid.

Een fase waarin veel activiteiten parallel worden uitgevoerd. Een goede planning en taakverdeling – vastgelegd en beheerd in de actielijst – zijn dan ook van groot belang.

Op te leveren producten	*Voor*	*Datum/Locatie*
1. Product 1.0 (in tweevoud)	Opdrachtgever Teamcoach	week 10 – 2/4 2008 (voor 13.00 uur)
2. Documentatie (in tweevoud)	Opdrachtgever Teamcoach	week 10 – 2/4 2008 (voor 13.00 uur)
3. Product presentatie	Opdrachtgever Teamcoach	week 10 – 3/4 2008 (volgens afspraak)
Nb. Na de presentatie geven de teamcoach en de opdrachtgever direct hun eerste feedback op de gepresenteerde producten en de wijze van presenteren. De teams krijgen tijdens de evaluatie en beoordeling gedetailleerde feedback over het product, de presentatie en de documentatie.		

1. Product versie 1.0
Het product versie 1.0 ***werkt***, is gevuld met content en wordt gedemonstreerd tijdens de product presentatie.

Randvoorwaarden
- Het ***product*** wordt op CD/DVD in een ***DVD verpakking*** met kaft, met begeleidende ***presentatie*** en ***handleiding*** ingeleverd. In geval van een website wordt de URL opgenomen in de begeleidende presentatie.
- Opleveren in tweevoud

Ad1. Begeleidende presentatie op CD/DVD
De CD/DVD bevat een presentatie (.ppt) met de volgende onderdelen:
- Titel van het product
- Klas, teamnaam, teambeeldmerk en namen van teamleden
- Omschrijving van de opdracht
- Omschrijving van het product & Unieke aspecten van het product
- Één of meerdere karakteristieke screenshots
- URL (in geval het product een website is)

Randvoorwaarden
- Verzorgd uitgevoerd en passend bij product
- Passend in en bij DVD verpakking
- Passende presentatie van informatie in tekst, diagrammen en/of beeld
- Geschreven in correct Nederlands

Ad2. Handleiding in DVD verpakking
Instructies voor het implementeren, beheren en gebruiken van het product voor de verschillende gebruikersgroepen. Afhankelijk van het product en organisatie:
- Technische documentatie voor onderhoud en implementatie
- Handleiding(en) voor dagelijks gebruik

Randvoorwaarden

- Uitvoering afhankelijk van product (bijv. eenmalige installatie-instructie, naslagwerk, tutorial, quick-reference card...)
- Verzorgd uitgevoerd en passend bij product en doelgroep en bij DVD verpakking
- Passende presentatie van informatie in tekst, diagrammen en/of beeld
- Geschreven in correct Nederlands
- Bevat teamnaam, klas en namen van teamleden
- Eventueel kan de handleiding ook op de CD/DVD gezet worden

2. Documentatie

De documentatie is een bijgewerkte versie van de in lesweek 4 opgeleverde ontwerp specificatie.

Randvoorwaarden

- 12 à 16 A4 (inclusief kaft, exclusief bijlagen)
- Opleveren in tweevoud
- Verzorgd uitgevoerd en passend in de teamstijl
- Passende presentatie van informatie in tekst, diagrammen en/of beeld
- Geschreven in correct Nederlands
- Bevat teamnaam, klas en namen en studentnummers van teamleden

3. Product presentatie

Het team presenteert ***het product*** aan de opdrachtgever en de teamcoach, waarbij het kernidee en de primaire functionaliteit (voor de verschillende gebruikersgroepen) overgebracht wordt.

Na de presentatie geven de teamcoach en de opdrachtgever direct hun eerste feedback op het gepresenteerde product en de wijze van presenteren. De teams krijgen tijdens de evaluatie en beoordeling in lesweek 10 van blok 3 gedetailleerde feedback over het product, de presentatie en de documentatie.

Na afloop van de presentatie bepaalt de teamcoach welke teams tot de voorselectie van de nominaties van de Golden Dot awards behoren. De Golden Dot awards is de jaarlijkse verkiezing van het beste projectwerk uit alle studiejaren.

Daarnaast wordt bekeken welke producten levensvatbaar zijn en geïmplementeerd kunnen worden (zie 'Implementatie en overdracht').

Randvoorwaarden

- 15 à 20 minuten inclusief demonstratie van het product
- Verzorgd uitgevoerd en passend bij de teamstijl en het product
- Geschreven in correct Nederlands
- Beamer en boxen zijn aanwezig (overige middelen moeten zelf verzorgen worden)

Ad. Implementatie en overdracht

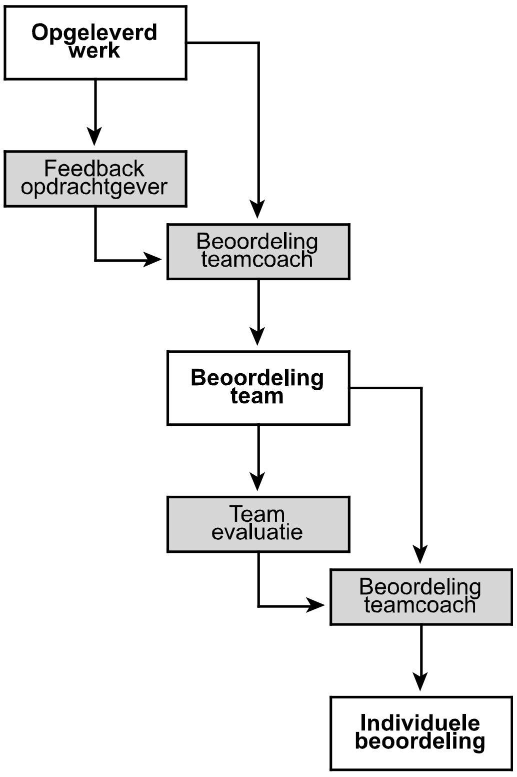
Een geslaagde implementatie van het product in de organisatie van de opdrachtgever op de systemen van de opdrachtgever is het ultieme doel van het projectwerk. Na afloop van de presentatie wordt door de opdrachtgever en de teamcoach bepaald welke teams een levensvatbaar product hebben ontworpen.

In overleg tussen die teams, de opdrachtgever en de teamcoach wordt bepaald hoe die producten geïmplementeerd worden. De implementatie van deze producten moet in ieder geval afgerond zijn voor 10 april (lesweek 1 van blok 4). Met teams waarvoor het implementeren en overdragen van het product meer werk is worden speciale afspraken gemaakt.

2.5 Teamevaluatie

Projectwerk is aan de ene kant leuk, leerzaam en uitdagend, maar aan de andere kant vaak ook lastig en soms frustrerend. In de hitte van het werk is het echter moeilijk om goed op al die aspecten te letten. Om het doorlopen proces goed te evalueren wordt na afloop van het tweede en het derde blok een teambeoordeling gehouden. Het doel van deze evaluatie is tweeledig:

1. De sterke en de verbeterpunten in ***het doorlopen proces en teamwerk*** onderkennen, om vervolgens te bedenken hoe sterke punten in volgende projecten behouden kunnen blijven en hoe beter met zwakke punten kan worden omgegaan.
2. Het evalueren van de eigen en elkaars bijdrage en het onderkennen van (op deelgebieden) ***individuele bovenmatige en ondermaatse prestaties***, om de teamcoach vervolgens te adviseren over aanpassing van de teambeoordeling voor individuele studenten.



Beoordelingproces

Het proces verloopt als volgt:

1. Projectwerk is teamwerk bij uitstek. De basis voor de beoordeling van het projectwerk is dan ook de beoordeling van het werk dat het ***team als geheel*** oplevert. De opdrachtgever adviseert de teamcoach bij het bepalen van deze beoordeling.
2. Vervolgens wordt op basis van de evaluatie van de ***individuele prestaties*** de beoordeling voor elke student bepaald. De uitkomsten van deze teamevaluatie neemt de teamcoach hierbij als advies mee.

   Nb. De ***teamcoach is verantwoordelijk*** voor zowel de beoordeling van het team als geheel als voor positieve of negatieve individuele aanpassingen van de beoordeling. De opdrachtgever en het team hebben daarbij een adviserende stem.

Tijdspad

Het projectwerk wordt in week 9 van het derde blok afgesloten met de presentatie van het product. De teamevaluatie wordt door de teamcoach afgenomen in lesweek 10 van het vierde blok.

1. Elke teamlid vult voorafgaande aan de teamevaluatie het formulier op de volgende pagina in.
2. De teamcoach bepaalt (samen met de opdrachtgever) voorafgaande aan de teamevaluatie de beoordeling van het team als geheel
3. Tijdens de teamevaluatie in de eerste week van het vierde blok bepaald het team voor elk teamlid een individueel advies voor de teamcoach.
4. Tijdens de teamevaluatie de teamcoach bepaalt voor elk teamlid de individuele beoordeling
5. Tijdens de teamevaluatie Indien nodig worden afspraken gemaakt met het team als geheel of individuele studenten in het bijzonder.

Teamevaluatie: Project blok 3

Stap 1. Persoonlijke gegevens

Klas/Team	Naam
Project	Teamcoach

Stap 2. Teamleden *Vul de namen van de teamleden (inclusief jezelf) in.*

Stem de nummering van de leden onderling af zodat alle formulieren gelijk zijn.

Teamlid 1	Teamlid 4
Teamlid 2	Teamlid 5
Teamlid 3	Teamlid 6

Stap 3. Beoordeling *van de prestatie van de teamleden (inclusief jezelf).*

Beoordeel onderstaande aspecten afzonderlijk en het geheel met:
***O**(nvoldoende), **V**(oldoende), **R**(uim) **V**(oldoende), **G**(oed) of **U**(itmuntend).*

Aspect	*Teamlid 1*	*Teamlid 2*	*Teamlid 3*	*Teamlid 4*	*Teamlid 5*	*Teamlid 6*
Taakuitvoering (kwantiteit)						
Taakuitvoering (kwaliteit)						
Betrokkenheid						
Collegialiteit						
Creativiteit						
Totaal						

Stap 4. Commentaar teamprestatie *Beste/zwakste punten van de teamprestatie.*

dit ging goed

dit kan beter

Stap 5. Persoonlijk commentaar *Beste/zwakste punten van je individuele prestatie.*

dit ging goed

dit kan beter

Stap 6. Commentaar *Bespreek het commentaar met je teamleden.*

Vul het oordeel per aspect en de belangrijkste opmerkingen over jezelf in de verzamelstaat in.

Aspect	***Oordeel***	***Commentaar***
Taakuitvoering (kwantiteit)		
Taakuitvoering (kwaliteit)		
Betrokkenheid		
Collegialiteit		
Creativiteit		
Totaal		

Stap 7. Ondertekening

datum & handtekening student	datum & handtekening docent

Deel 3: Reader

Inspiratie, theorie, voorbeelden en achtergrond

Usability testing on 10 cents a day

uit: Don’t Make me Think, Steve Krug, ISBN 0-7897-2310-7

Keeping testing simple – so you do enough of it
Why didn't we do this sooner?

What everyone says at some point during the first usability test of their web site

About once a month, I get one of these phone calls:

As soon as I hear "launching in two weeks" (or even "two months”) and "usability testing" in the same sentence, I start to get that old fireman-headed-into-the-burning-chemical-factory feeling. because I have a pretty good idea of what's going on.

If it's two weeks, then it's almost certainly a request for a disaster check. The launch is fast approaching and everyone's getting nervous, and someone finally says, "Maybe we better do some usability testing."

If it's two months, then odds are that what they want is to settle some ongoing internal debates-usually about something very specific like color schemes. Opinion around the office is split between two different designs; some people like the sexy one, some like the elegant one. Finally someone with enough clout to authorize the expense gets tired of the arguing and says, "All right, let's get some testing done to settle this."

And while usability testing will sometimes settle these arguments, the main thing it usually ends up doing is revealing that the things they were arguing about aren't all that important. People often test to decide which color drapes are best, only to learn that they forgot to put windows in the room. For instance, they might discover that it doesn't make much difference whether you go with the horizontal navigation bar or the vertical menus if nobody understands the value proposition of your site.
Sadly, this is how most usability testing gets done: too little, too late, and for all the wrong reasons.

Repeat after me: Focus groups are not usability tests.
Sometimes that initial phone call is even scarier:
When the last-minute request is for a focus group, it's usually a sign that the request originated in Marketing. When Web sites are being designed, the folks in Marketing often feel like they don't have much clout. Even though they're the ones who spend the most time trying to figure out who the site's audience is and what they want, the designers and developers are the ones with most of the hands-on control over how the site actually gets put together.

Ed Grimley at XYZ Corp gave me your name.
We're launching our site in two weeks and we want to do some usability testing.
...two weeks?
...we're launching our site in two weeks and we want to do some *focus group testing*.
Focus group testing?

As the launch date approaches, the Marketing people may feel that their only hope of sanity prevailing is to appeal to a higher authority: research. And the kind of research they know is focus groups. I often have to work very hard to make clients understand that what they need is usability testing, not focus groups. Here's the difference in a nutshell:

– ***In a focus group***, a small group of people (usually 5 to 8) sit around a table and react to ideas and designs that are shown to them. It's a group process, and much of its value comes from participants reading to each other's opinions. Focus groups are good for quickly getting a sampling of user's opinions and feelings about things.
– ***In a usability test***, one user at a time is shown something (whether it's a Web site, a prototype of a site, or some sketches of individual pages) and asked to either (a) figure out what it is, or (b) try to use it to do a typical task.

Focus groups can be great for determining what your audience wants, needs, and likes-in the abstract. They're good for testing whether the idea behind the site makes sense and your value proposition is attractive. And they can be a good way to test the names you're using for features of your site, and to find out how people feel about your competitors.
But they're not good for learning about whether your site works and how to improve it.
The kinds of things you can learn from focus groups are the things you need to learn early on, before you begin designing the site. Focus groups are for EARLY in the process. You can even run them late in the process if you want to do a reality check and fine-tune your message, but don't mistake them for usability testing. They wont tell you whether people can actually use your site.

Several true things about testing
Here are the main things I know:

– ***If you want a great site, you’ve got to test.*** After you've worked on a site for even a few weeks, you can't see it freshly anymore. You know too much. The only way to find out if it really works is to test it.
Testing reminds you that not everyone thinks the way you do, knows what you know, uses the Web the way you do.
I used to say that the best way to think about testing was that it was like travel: a broadening experience. It reminds you how different - and the same - people are, and gives you a fresh perspective on things.
But I finally realized that testing is really more like having friends visiting from out of town. Inevitably, as you make the tourist rounds with them, you see things about your home town that you usually don't notice because you're so used to them. And at the same time, you realize that a lot of things that you take for granted aren't obvious to everybody.
– ***Testing one user is 100 percent better than testing none.*** Testing always works. Even the worst test with the wrong user will

show you things you can do that will improve your site.

- ***Testing one user early in the project is better than testing 50 near the end.*** Most people assume that testing needs to be a big deal. But if you make it into a big deal, you won't do it early enough or often enough to get the most out of it. A simple test early-while you still have time to use what you learn from it-is almost always more valuable than a sophisticated test later. Part of the conventional wisdom about Web development is that it's very easy to go in and make changes. The truth is, it turns out that it's not that easy to make changes to a site once it's in use. Some percentage of users will resist almost any kind of change, and even apparently simple changes often turn out to have far-reaching effects, so anything you can keep from building wrong in the first place is gravy.
- ***The importance of recruiting representative users is overrated.*** It's good to do your testing with people who are like the people who will use your site, but it's much more important to test early and often. My motto – as you'll see – is "Recruit loosely, and grade on a curve."
- ***The point of testing is not to prove or disprove something. It's to inform your judgment.*** People like to think, for instance, that they can use testing to prove whether navigation system "a" is better than navigation system "b", but you can't. No one has the resources to set up the kind of controlled experiment you'd need. What testing can do is provide you with invaluable input which, taken together with your experience, professional judgment, and common sense, will make it easier for you to choose wisely-and with greater confidence--between "a" and "b."
- ***Testing is an iterative process.*** Testing isn't something you do once. You make something, test it, fix it, and test it again.
- ***Nothing beats a live audience reaction.*** One reason why the Marx Brothers' movies are so wonderful is that before they started filming they would go on tour on the vaudeville circuit and perform scenes from the movie, doing five shows a day, improvising constantly and noting which lines got the best laughs. Even after they'd settled on a line, Groucho would insist on trying slight variations to see if it could be improved.

Lost our lease, going-out-of-business- sale usability testing

Usability testing has been around for a long time, and the basic idea is pretty simple: If you want to know whether your software or your Web site or your VCR remote control is easy enough to use, watch some people while they try to use it and note where they run into trouble. Then fix it, and test it again.

In the beginning, though, usability testing was a very expensive proposition. You had to have a usability lab with an observation room behind a one-way mirror, and at least two video cameras 50 you could record the users' reactions and the thing they were using. You had to recruit a lot of people so you could get results that were statistically significant. It was Science. It cost $20 to $50,000 a shot. It didn't happen very often.

But in 1989 Jakob Nielsen wrote a paper titled "Usability Engineering at a Discount" and pointed out that it didn't have to be that way. You didn't need a usability lab, and you could achieve the same results with a lot fewer users.

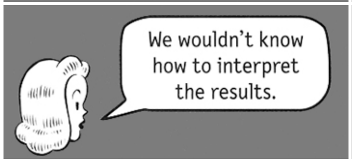
The idea of discount usability testing was a huge step forward. The only problem is that a decade later most people still perceive testing as a big deal, hiring someone to conduct a test still costs $5 to $15,000, and as a result it doesn't happen nearly often enough.

What I'm going to commend to you in this chapter is something even more drastic: Lost our lease, going-out-of-business-sale usability testing.

I'm going to try to explain how to do your own testing when you have no money and no time. If you can afford to hire a professional to do your testing, by all means do it –but don't do it if it means you'll do less testing.

	TRADITIONAL TESTING	LOST-OUR-LEASE TESTING
NUMBER OF USERS PER TEST	Usually eight or more to justify the set-up casts	Three or four
RECRUITING EFFORT	Select carefully to match target audience	Grab some people. Almost anybody who uses the Web will do.
WHERE TO TEST	A usability lab, with an observation room and a one-way mirror	Any office or conference room
WHO OOES THE TESTING	An experienced usability professional	Any reasonably patient human being
AOVANCE PLANNING	Tests have to scheduled weeks in advance to reserve a usability lab and allow time for recruiting	Tests can be done almost any time, with little advance scheduling
PREPARATION	Draft, discuss, and revise a test protocol	Decide what you're going to show
WHAT/WHEN DO YOU TEST?	Unless you have a huge budget, put all your eggs in one basket and test once when the site is nearly complete	Run small tests continually throughout the development process
COST	$5,000 to $15,000 (or more)	About $300 (a $50 to $100 stipend for each user and $20 for three hours of videotape)
WHAT HAPPENS AFTERWARDS	Al 20-page written report appears a week later, then the development team meets to decide what changes to make	Each observer writes one page of notes the day of the test. The development team can debrief the same day

THE TOP FIVE PLAUSIBLE EXCUSES FOR NOT TESTING WEB SITES

	It's true that most Web development schedules seem to be based on the punch line from a Dilbert cartoon. If testing is going to add to everybody's to-do list, if you have to adjust development schedules around tests and involve key people in preparing for them, then it won't get done. That's why you have to make testing as small a deal as possible. Done right, it will save time, because you won't have to (a) argue endlessly, and (b) redo things at the end.
	Forget $5 to $15,000. If you can convince someone to bring in a camcorder from home you'll only need to spend about $300 for each round of tests.
	The least-known tact about usability testing is that it's incredibly easy to do. Yes, same people will be better at it than others, but I've never seen a usability test fail to produce useful results, no matter how poorly it was conducted.
	You don't need one. All you really need is a room with a desk, a computer, and two chairs where you won't be interrupted.
	It's true, the trickiest part of usability testing is making sure you draw the right conclusions from what you see. We'll cover that in the next chapter.

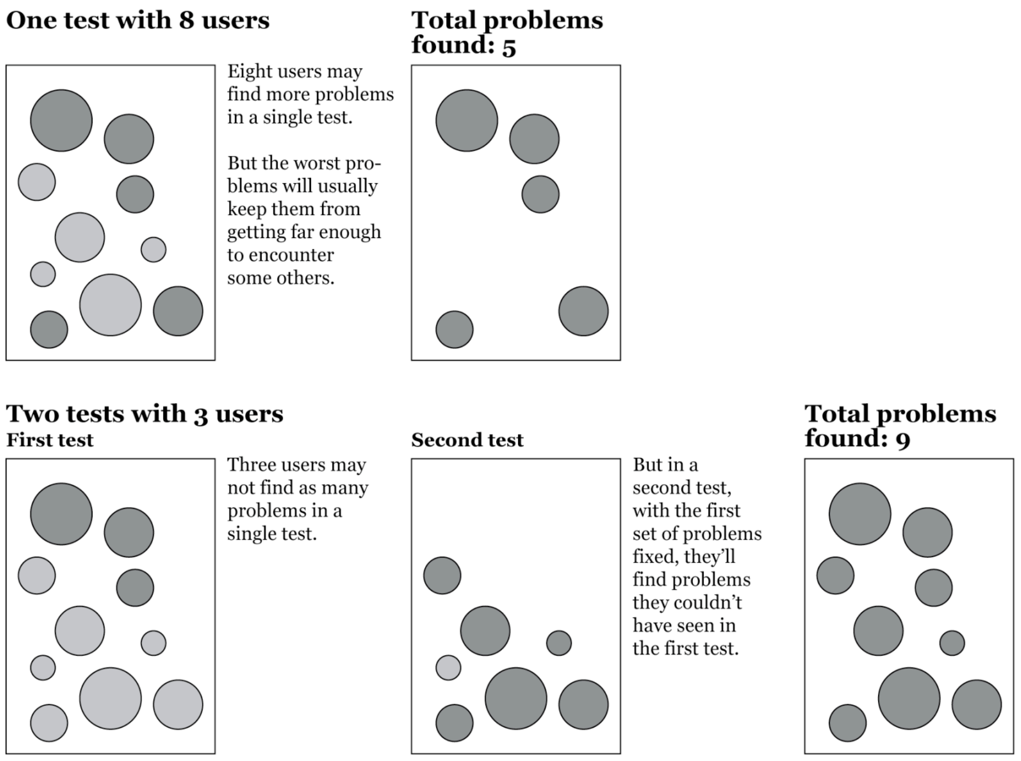
How many users should you test?
In most cases, I tend to think the ideal number of users for each round of testing is three, or at most four.
The first three users are very likely to encounter all of the most significant problems: and it's much more important to do more rounds of testing than to wring everything you can out of each round. Testing only three users helps ensure that you will do another round soon.
Also, since you will have fixed the problems you uncovered in the first round, in the next round it's likely that all three users will uncover a new set of problems, since they won't be getting stuck on the first set of problems.
Testing only three or four users also makes it possible to test and debrief in the same day, so you can take advantage of what you've learned right away.
Also, when you test more than four at a time, you usually end up with more notes than anyone has time to process-many of them about things that are really "nits," which can actually make it harder to see the forest for the trees.
It's better to stay focused on the biggest problems, fix them, and then test again as soon as possible.

Recruit loosely and grade on a curve
When people decide to test, they often spend a lot of time trying to recruit users who they think will precisely reflect their target audience-for instance, male accountants between the ages of 25 and 30 with one to three years of computer experience who have recently purchased expensive shoes.
The best-kept secret of usability testing is the extent to which it doesn't much matter who you test.
For most sites, all you really need are people who have used the Web enough to know the basics.
If you can afford to hire someone to recruit the participants for you and it won't reduce the number of rounds of testing that you do, then by all means be as specific as you want. But if finding the ideal user means you're going to do fewer tests, I recommend a different approach: *Take anyone you can get (within limits) and grade on a curve*.
In other words, try to find users who reflect your audience, but don't get hung up about it. Instead, try to make allowances for the differences between the people you test and your audience.
I favor this approach for three reasons:

- ***We're all beginners under the skin.*** Scratch an expert and you'll often find someone who's muddling through-just at a higher level.
- ***It's usually not a good idea to design a site so that only your target audience can use it.*** If you design a site for accountants using terminology that you think all accountants will understand, what you'll probably discover is that a small but not insignificant number of accountants won't know what you're talking about. And in most cases, you need to be addressing novices as well as experts anyway, and if your grandmother can use it, an expert can.
- ***Experts are rarely insulted by something that is clear enough for beginners.*** Everybody appreciates clarity. (true clarity, that is, and not just something that's been "dumbed down.")

The exceptions:

- ***If your site is going to be used almost exclusively by one type of user and it's no harder to recruit from that group***, then do it. For instance, if your audience will be almost entirely women, then by all means test just women.
- ***If your audience is split between clearly defined groups with very divergent interests and needs***, then you need to test users from each group at least once. For instance, if you're building a university site, for at least one round of testing you want to recruit two students, two professsors, two high school seniors, and two administrators. But for the other rounds, you can choose any mix.



One test with 8 users

Eight users may find more problems in a single test.

But the worst problems will usually keep them from getting far enough to encounter some others.

Total problems found: 5

Two tests with 3 users

First test

Three users may not find as many problems in a single test.

Second test

But in a second test, with the first set of problems fixed, they'll find problems they couldn't have seen in the first test.

Total problems found: 9

– ***If using your site requires specific domain knowledge*** (e.g., a currency exchange site for money management professionals), then you need to recruit people with that domain knowledge for at least one round of tests. But don't do it for every round if it will reduce the number of tests you do.

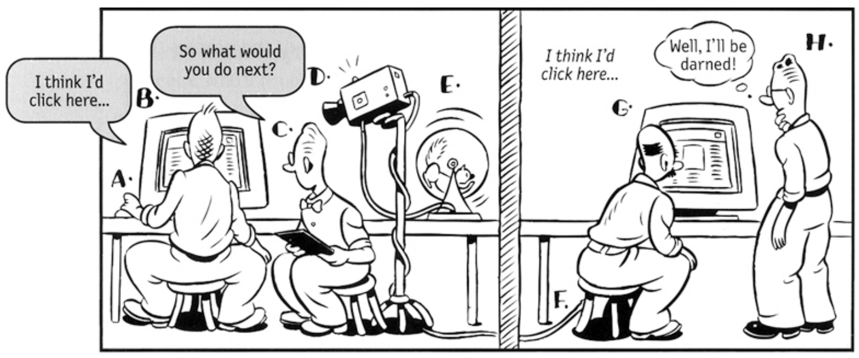
When you're recruiting:

– ***Offer a reasonable incentive.*** Typical stipends for a one-hour test session range from $50 for "average" Web users to several hundred dollars for professionals from a specific domain, like cardiologists for instance. I like to offer people a little more than the going rate, since (a) it makes it clear that I value their opinion, and (b) people tend to show up on time, eager to participate. Remember, even if the session is only 30 minutes, people usually have to block out another hour for travel time. Also, I'd rather have people who are curious about the process than people who are desperate for the money.

– ***Keep the invitation simple.*** "We need to have a few people look at our Web site and give us some feedback. It's very easy, and would take about forty-five minutes to an hour. And you'll be paid $xx for your time."

– ***Avoid discussing the site (or the organization behind the site) beforehand.*** You want their first look to tell you whether they can figure out what it is from a standing start. (Of course, if they're coming to your office, they'll have a pretty good idea whose site it is.)

– ***Don't be embarrassed to ask friends and neighbors.*** You don't have to feel like you're imposing if you ask friends or neighbors to participate. Most people enjoy the experience. It's fun to have someone take your opinion seriously and get paid for it, and they often learn something useful that they didn't know about the Web or computers in general.


I think I'd click here...
So what would you do next?
A.
B.
C.
D.
E.
I think I'd click here...
Well, I'll be darned!
F.
G.
H.

Where do you test?

All you really need is an office or conference room with two chairs, a PC or Mac (with an Internet connection, if you're testing a live site), a camcorder, and a tripod.

I recommend running a long cable from the camcorder to a TV in another office – or even a cubicle-nearby and encouraging everyone on the development team to come and watch.

The camcorder needs to record what the user sees (the computer screen or the designs on paper, depending on what you're testing) and what the user and the facilitator say. In most cases, you'll never go back and look at the videotapes, but they're good to have anyway, particularly to show to team members who want to observe but can't.

You can buy the camcorder, TV, cable, and tripod for less than $600. But if your budget won't stretch that far, you can probably twist somebody's arm to bring in a camcorder from home on test days.

Who should do the testing?
Almost anyone can facilitate a usability test; all it really takes is the courage to try it. With a little practice, most people can get quite good at it.
Try to choose someone who tends to be patient, calm, empathetic, a good listener, and inherently fair. Don't choose someone whom you would describe as "definitely not a people person" or "the office crank."

Who should observe?
Anybody who wants to. It's a good idea to encourage everyone - team members, people from marketing and business development, and any other stakeholders - to attend. If you can, try to get senior management to at least drop by; they'll often become fascinated and stay longer than they planned.

What do you test, and when do you test it?
The table below shows the different kinds of testing you should do at each phase of Web development.
Before you even begin designing your site, you should be testing comparable sites. They may be actual competitors, or they may be sites that are similar in style, organization, or features to what you have in mind. Use them yourself, then watch one or two other people use them and see what works and what doesn't. Many people overlook this step, but it's invaluable-like having someone build a working prototype for you for free.
If you've never conducted a test before testing comparable sites, it will give you a pressure-free chance to get the hang of it. It will also give you a chance to develop a thick skin. The first few times you test your own site, it's hard not to take it personally when people don't get it. Testing someone else's site first will help you see how people react to sites and give you a chance to get used to it. Since the comparable sites are "live," you can do two kinds of testing: "Get it" testing and key tasks.

– ***"Get it" testing*** is just what it sounds like: show them the site, and see if they get it –do they understand the purpose of the site, the value proposition, how it's organized, how it works, and so on.
– ***Key task testing*** means asking the user to do something, then watching how well they do. As a rule, you'll always get more revealing results if you can find a way to observe users doing tasks that they have a hand in choosing. It's much better, for instance, to say "Find a book you want to buy, or a book you bought recently' than "Find a cookbook for under $14" When people are doing made-up tasks, they have no emotional investment in it, and they can't use as much of their personal knowledge.

As you begin designing your own site, it's never too early to start showing your design ideas to users, beginning with your first rough sketches. Designers are often reluctant to show work in progress, but users may actually feel freer to comment on something that looks unfinished, since they know you haven't got as much invested in it and it's still subject to change. Also, since it's not a polished design, users won't be distracted by details of implementation and they can focus on the essence and the wording.
Later, as you begin building parts of the site or functioning prototypes, you can begin testing key tasks on your own site. I also recommend doing what I call Cubicle tests: Whenever you build a new kind of page – particularly forms – you should print the page out and show it to the person in the next cubicle and see if they can make sense out of it. This kind of informal testing can be very efficient, and eliminate a lot of potential problems.

	PLANNING	ROUGH SKETCHES	PAGE DESIGNS	PROTOTYPE	FIRST USABLE VERSION	"CUBLICLE TESTS"
WHAT TO TEST	Competitors' sites	Sketch of Home page Names of top level categories and site features	Home page Second-level page templates Content page templates	As much as you have working	As much as you have working	Each unique page
FORMAT	Live site	Paper	Paper	HTML prototype	Live site	HTML page
HOW TO TEST	"Get it" Key tasks	"Get it" Names of things	"Get it" Basic Navigation	"Get it" Key tasks	"Get it" Key tasks	Key tasks
WHAT YOU'RE LOOKING FOR	What do they like/love? How does it fit into their lives? What works well? How hard is it to de key tasks?	Do they get the point of the site? Does it seem like what they need?	Do they get the point of the site? Do they get the navigation? Can they guess where to find things?	Do they still get it? Cab they accomplish the key tasks?	Do they still get it? Can they accomplish the key tasks?	Can they accomplish the key tasks?
SESSION LENGTH	1 hour	15-20 minutes	15-20 minutes	45 min – 1 hour	1 hour	5 minutes per page
# OF TETS	1	1-3	1-3	1-3	1-3	1 per page
	TOTAL BUDGET: 13 TESTS X 3 USERS PER TEST X $100 PER USER = $3900					

Usability testing the movie

uit: Don’t Make me Think, Steve Krug, ISBN 0-7897-2310-7

I don't like the colors.

What you can count on at least one user saying in every usability test

This chapter explains how to conduct a test if you're the facilitator and what to look for if you're an observer.

If you've never conducted a usability test, the main thing you need to know is that you should just relax, because there's not much to it. Your responsibility is to tell the users what you want them to do, to encourage them to think out loud, to listen carefully to what they have to say, and to protect them. Here's a list of the things to keep in mind. If you read this list and the sample session that follows it, you'll be ready to start testing.

- ***Try the test yourself first.*** The day before the test, try doing whatever you're going to ask the test participants to do and make sure that you can do it in the time allotted. Make sure that whatever pages you're testing are accessible from the computer you’ll be using, and that you have any passwords you’ll need.
- ***Protect the participants.*** It's your responsibility to prevent any damage to your test users' self-esteem. Be nice to them. If they get stuck, don't let them get too frustrated. Be sure to pat them on the back (figuratively), and thank them (sincerely) when you're done. Let them know that their participation has been very helpful-exactly what you needed.
- ***Be empathetic.*** Be kind, patient, and reassuring. Make it clear that you know they're not stupid.
- ***Try to see the thought balloons forming over their heads.*** The main thing you're trying to do is observe their thought process. Whenever you're not sure what they're thinking, ask them. If the user has been staring at the screen for ten seconds, ask, "What are you looking at?" or "What are you thinking?" You're trying to see what their expectation is at each step and how close the site comes to matching that expectation. When they're ready to click, ask what they expect to see. After they click, ask if the result was what they expected.
- ***Don’t give them hints about what to do.*** It's a lot like being a therapist. If they say, "I’m not sure what to do next. You should say, "What do you think you should do?" or "What would you do if you were at home?"
- ***Keep your instructions simple.*** There aren't very many, so you'll learn them quickly.

  "Look around the page and tell me what you think everything is and what you would be likely to click on.”

  "Tell me what you would click on next and what you expect you would see then."

  "Try to think out loud as much as possible.”

  Don't be afraid to keep repeating them; it will be more boring to you than to the user.
- ***Probe, probe, probe.*** You have to walk a delicate line between distracting or influencing the users and finding out what they're really thinking, which they may not know themselves.
  For instance, when a user says, "I like this page" you always want to ask a leading question like "What do you like best about it?" If this produces "Well, I like the layout" then you need to follow with "What appeals to you about the layout?" You're looking for specifics, not because the specifics themselves are necessarily important but because eliciting them is the only way you can be sure you understand what the user is really reacting to.
- ***Don’t be afraid to improvise.*** For instance, if the first two users get hopelessly stuck at the same point and it's obvious what the problem is and how to fix it, don't make the third user struggle with it needlessly. As soon as he encounters the problem, explain it and let him go on to something more productive.
- ***Don’t be disappointed if a user turns out to be inexperienced or completely befuddled.*** You can often learn more by watching a user who doesn't get it than one who does. Because more experienced users have better coping strategies for "muddling through,“ you may not even notice that they don't get it.
- ***Make some notes after each session.*** Always take a few minutes right after each test session to jot down the main things that struck you. If you don't do it before you start the next test, it will be very hard to remember what they were.

A sample test session

Here's an annotated excerpt from a typical-but imaginary-test session. The site is real, but it has since been redesigned. The participant's name is Janice, and she's about 25 years old.

Introduction	
Hi, Janice. My name is Steve Krug, and I'm going to be walking you through this session.	This whole first section is the script that I use when I conduct tests.
You probably already know, but let me explain why we've asked you to come here today. We're testing a Web site that we're working on so we can see what it's like for actual people to use it. I want to make it clear right away that we're testing the site, not you. You can't do anything wrong here. In fact, this is probably the one place today where you don't have to worry about making mistakes. We want to hear exactly what you think, so please don't worry that you're going to hurt our feelings. We want to improve it, so we need to know honestly what you think. As we go along, I'm going to ask you to think out loud, to tell me what's going through your mind. This will help us.	I always have a copy in front of me, and I don't hesitate to read from it, but I find it's good to ad lib a little, even if it means making mistakes. When the users see that I'm comfortable making mistakes, it helps take the pressure off them.
If you have questions, just ask. I may not be able to answer them right away, since we're interested in how people do when they don't have someone sitting next to them, but I will try to answer any questions you still have when we're done. We have a lot to do, and I'm going to try to keep us moving, but we'll try to make sure that it's fun, too.	It's important to mention this, because it will seem rude not to answer their questions as you go along. You have to make it clear before you start that (a) it's nothing personal, and (b) you'll try to answer them at the end if they still want to know.
You may have noticed the camera. With your permission, we're going to videotape the computer screen and what you have to say. The video will be used only to help us figure out how to improve the site, and it won't be seen by anyone except the people working on the project. It also helps me, because I don't have to take as many notes. There are also some people watching the video in another room.	At this point, most people will say something like, I'm not going to end up on America's Funniest Home Videos, am I?"
If you would, I'm going to ask you to sign something for us. It simply says that we have your permission to tape you, but that it will only be seen by the people working on the project. It also says that you won't talk to anybody about what we're showing you today, since it hasn't been made public yet. Do you have any questions before we begin? No. I don't think so.	Give them the release and non-disclosure agreement to sign. It should be as short as possible and written in plain English.
Background questions	
Before we look at the site, I'd like to ask you just a few quick questions. First, what's your occupation? I'm a router. I've never heard of that before. What does a router do, exactly? Not much. I take orders as they come in, and send them to the right office. Good. Now, roughly how many hours a week would you say you spend using the Internet, including email?	I find it's good to start with a few questions to get a feel for who they are and how they use the Internet, It gives them a chance to loosen up a little and gives you a chance to show that you're going to be listening attentively to what they say-and that there are no wrong or right answers. Don't hesitate to admit your ignorance about anything. Your role here is not to come across as an expert, but as a good listener.
Oh, I don't know. Probably an hour a day at work, and maybe four hours a week at home. Mostly that's on the weekend. I'm too tired at night to bother. But I like playing games sometimes. How do you spend that time? In a typical day, for instance, tell me what you do, at work and at home.	Notice that she's not sure how much time she really spends on the Internet. Most people aren't. Don't worry. Accurate answers aren't important here. The main point here is just to get her talking and thinking about how she uses the Internet and to give you a chance to gauge what kind of user she is.

Well, at the office I spend most of my time checking email. I get a lot of email, and a lot of it's junk but I have to go through it anyway. And sometimes I have to research something at work.

Do you have any favorite Web sites?

Yahoo, I guess, l like Yahoo, and I use it a1J the time. And something called Snakes.com, because I have a pet snake.

Really? What kind of snake?

A python. He's about four feet long, but he should get to be eight or nine when he's fully grown.

Wow. OK, now, finally, have you bought anything on the Internet? How do you feel about buying things on the Internet?

Don't be afraid to digress and find out more about the user, as long as you come back to the topic before long.

I've bought some things recently. I didn't do it for a long time, but on1y because I couldn't get things delivered. It was hard to get things delivered, because I'm not home during the day. But now one of my neighbors is home al1 the time, so I can.

And what have you bought?

Well, I ordered a raincoat from L.L. Bean, and it worked out much better than I thought it would. It was actually pretty easy.

OK. We're done with the questions, and we can start looking at things.

OK, I guess.

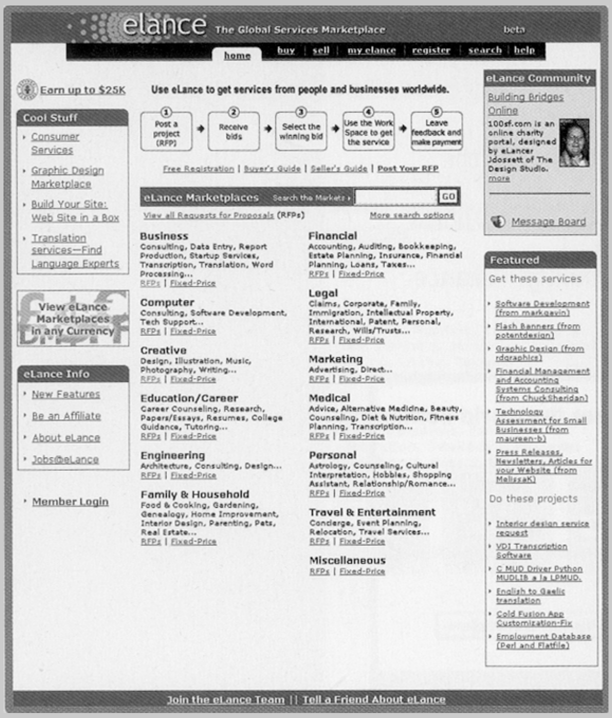
Reactions to the home page

First, I'm just going to ask you to look at this page and tell me what you think it is, what strikes you about it, and what you think you would click on first.

For now, don't actually click on anything. Just tell me what you would click on.

And again, as much as possible, it will help us if you can try to think out loud so we know what you're thinking about.

The browser has been open, but minimized. At this point, I reach over and click to maximize it.



elance The Global Services Marketplace beta

home | buy | sell | my elance | register | search | help

Earn up to $25K

Use eLance to get services from people and businesses worldwide.

1 Post a project (RFP) → 2 Receive bids → 3 Select the winning bid → 4 Use the Work Space to get the service → 5 Leave feedback and make payment

Free Registration | Buyer's Guide | Seller's Guide | Post Your RFP

Cool Stuff
- Consumer Services
- Graphic Design Marketplace
- Build Your Site: Web Site in a Box
- Translation services—Find Language Experts

View eLance Marketplaces in any Currency

eLance Info
- New Features
- Be an Affiliate
- About eLance
- Jobs@eLance

- Member Login

eLance Marketplaces Search the Markets › GO

View all Requests for Proposals (RFPs) More search options

Business
Consulting, Data Entry, Report Production, Startup Services, Transcription, Translation, Word Processing...
RFPs | Fixed-Price

Computer
Consulting, Software Development, Tech Support...
RFPs | Fixed-Price

Creative
Design, Illustration, Music, Photography, Writing...
RFPs | Fixed-Price

Education/Career
Career Counseling, Research, Papers/Essays, Resumes, College Guidance, Tutoring...
RFPs | Fixed-Price

Engineering
Architecture, Consulting, Design...
RFPs | Fixed-Price

Family & Household
Food & Cooking, Gardening, Genealogy, Home Improvement, Interior Design, Parenting, Pets, Real Estate...
RFPs | Fixed-Price

Financial
Accounting, Auditing, Bookkeeping, Estate Planning, Insurance, Financial Planning, Loans, Taxes...
RFPs | Fixed-Price

Legal
Claims, Corporate, Family, Immigration, Intellectual Property, International, Patent, Personal, Research, Wills/Trusts...
RFPs | Fixed-Price

Marketing
Advertising, Direct...
RFPs | Fixed-Price

Medical
Advice, Alternative Medicine, Beauty, Counseling, Diet & Nutrition, Fitness Planning, Transcription...
RFPs | Fixed-Price

Personal
Astrology, Counseling, Cultural Interpretation, Hobbies, Shopping Assistant, Relationship/Romance...
RFPs | Fixed-Price

Travel & Entertainment
Concierge, Event Planning, Relocation, Travel Services...
RFPs | Fixed-Price

Miscellaneous
RFPs | Fixed-Price

eLance Community
Building Bridges Online
100sf.com is an online charity portal, designed by eLancer Jdossett of The Design Studio.
more

Message Board

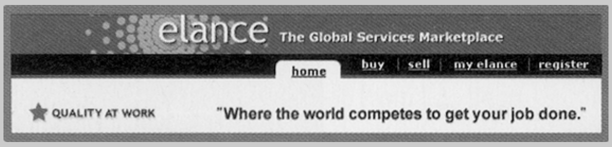
Featured
Get these services
- Software Development (from markgavin)
- Flash Banners (from potentdesign)
- Graphic Design (from rdgraphics)
- Financial Management and Accounting Systems Consulting (from ChuckSheridan)
- Technology Assessment for Small Businesses (from maureen-b)
- Press Releases, Newsletters, Articles for your Website (from MelissaK)

Do these projects
- Interior design service request
- VDI Transcription Software
- C MUD Driver Python MUDLIB a la LPMUD.
- English to Gaelic translation
- Cold Fusion App Customization-Fix
- Employment Database (Perl and Flatfile)

Join the eLance Team || Tell a Friend About eLance

Well, I guess the first thing I notice is that I like the color. I like the shade of orange, and I like the little picture of the sun [at the top of the page, in the eLance logo].

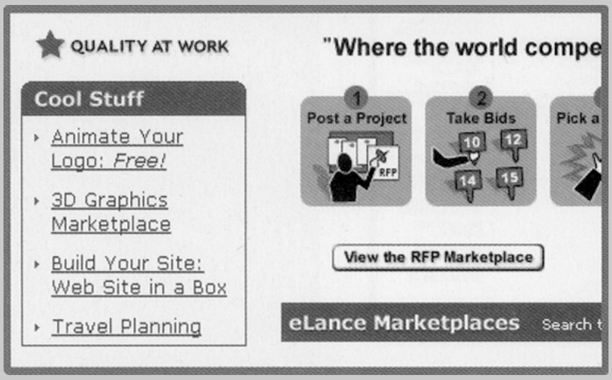
Let's see. [Reads.] "The global services market." "Where the world comes to get your job done."


elance The Global Services Marketplace
home
buy | sell | my elance | register
QUALITY AT WORK

"Where the world competes to get your job done."

I don't know what that means. I have no idea.

"Animate your logo free." [looking at the Cool Stuff section on the left.] "3D graphics marketplace." "eLance community." "eLance marketplace."



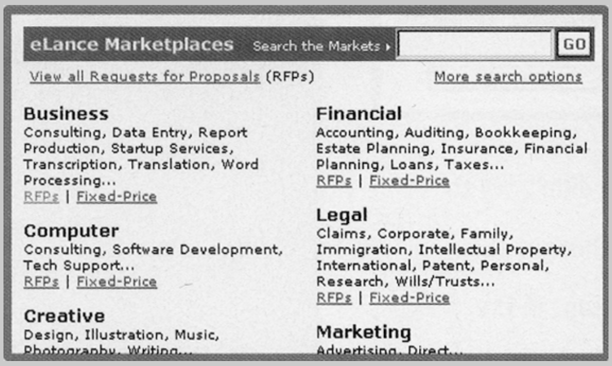
In an average test, it's just as likely that the next user will say that she hates this shade of orange and that the drawing is too simplistic. Don't get too excited by individual reactions to site aesthetics.

There's a lot going on here. But I have no idea what any of it is.

If you had to take a guess, what do you think it might be?

Well, it seems to have something to do with buying and selling...something.

[Looks around the page again.] Now that I look at the list down here [the Yahoo-style category list halfway down the page], I guess maybe it must be services. Legal, financial, creative...they all sound like services.



Business
Consulting, Data Entry, Report Production, Startup Services, Transcription, Translation, Word Processing...
RFPs | Fixed-Price

Computer
Consulting, Software Development, Tech Support...
RFPs | Fixed-Price

Creative
Design, Illustration, Music, Photography, Writing...

Financial
Accounting, Auditing, Bookkeeping, Estate Planning, Insurance, Financial Planning, Loans, Taxes...
RFPs | Fixed-Price

Legal
Claims, Corporate, Family, Immigration, Intellectual Property, International, Patent, Personal, Research, Wills/Trusts...
RFPs | Fixed-Price

Marketing
Advertising, Direct...

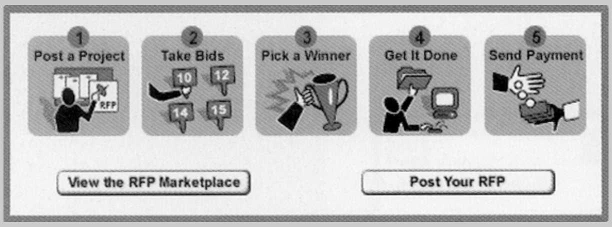
This user is doing a good job of thinking out loud on her own. If she wasn't, this is where I'd start asking her." What are you thinking?"

So I guess that's what it is. Buying and selling services. Maybe like some kind of online Yellow Pages.

OK. Now, if you were at home, what would you click on first?

I guess I'd click on that 30 graphics thing. I'm interested in 30 graphics.

Before you click on it, I have one more question. What about these pictures near the top of the page-the ones with the numbers? What did you make of them?



> I noticed them, but I really didn't try to figure them out. I guess I thought they were telling me what the steps in the process would be.

I ask this question because the site's designers think most users are going to start by clicking on the pictures of the five steps, and that everyone will at least look at them.

Any reason why you didn't pay much attention to them?

> No. I guess I just I just wasn't ready to start the process yet. I didn't know if I wanted to use it yet. I just wanted to look around first.

OK. Great.

At this point, I would let her explore for a few minutes, following whatever links she's interested in and encouraging her to think out loud the whole time.

Testing a task

OK, now we're going to try something else. Can you think of something you might want to post as a project if you were using this site?

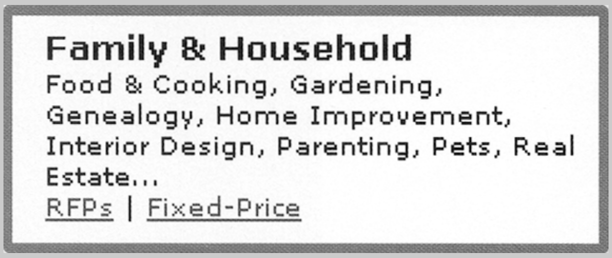
> Hmm. Let me think. I think I saw "Home Improvement" there somewhere. We're thinking of building a deck. Maybe I would post that.

So if you were going to post the deck as a project, what would you do first?

> I guess I'd click on one of the categories down here. I think I saw home improvement. [Looks.] There it is, under "Family and Household."

So what would you do?

> Well, I'd click.... [Hesitates, looking at the two links under " Family and Household."]



Family & Household
Food & Cooking, Gardening, Genealogy, Home Improvement, Interior Design, Parenting, Pets, Real Estate...
RFPs | Fixed-Price

After a few minutes of "free range clicking," I give her a task to perform so we can see whether she can use the site for its intended purpose. Whenever possible. I like to let the user have some say in choosing the task.

> Well, now I'm not sure what to do. I can't click on Home Improvement, so it looks like I have to click on either "RFPs" or "Fixed- Price," But I don't know what the difference is.
>
> Fixed price I sort of understand; they'll give me a quote, and then they have to stick to it. But I'm not sure what RFPs is.

As it turns out, she's mistaken. Fixed-price (in this case) means services available for a fixed hourly rate, while an RFP (or Request for Proposal) is actually the choice that will elicit quotes. This is the kind of misunderstanding that often surprises the people who built the site.

Well, which one do you think you'd click on?

> Fixed price, I guess.

Why don't you go ahead and do it?

From here on, I just watch while she tries to post a project, Jetting her continue until either (a) she finishes the task,(b) she gets really frustrated, or (c) we're not learning anything new by watching her try to muddle through.

What to do if you're observing
Being an observer at a usability test is a cushy job. All you have to do is listen, watch closely, keep an open mind, and take notes. Here are the types of things you're looking for:

- ***Do they get it?*** Without any help, can the users figure out what the site or the page is, what it does, and where to start?
- ***Can they find their way around?*** Do they notice and understand the site's navigation? Does your hierarchy-and the names you're using for things-make sense to them?
- ***Head slappers.*** You'll know these when you see them: the user will do something, or not do something, and suddenly everyone who's observing the session will slap his or her forehead and say, "Why didn't we think of that?" or "Why didn't we ever notice that?" These are very valuable insights.
- ***Shocks.*** These will also make you slap your head, but instead of saying "Why didn't we notice that?" you'll say, "How could she [the user] not notice that?" or "How could she not understand that?" For instance, you might be shocked when someone doesn't notice that there is a menu bar at the top of each page, or doesn't recognize the name of one of your company's products. Unlike head slappers, the solution to shocks won't always be obvious and they may send you back to the drawing board.
- ***Inspiration.*** Users will often suggest a solution or the germ of a solution to a problem that you've struggled with for a long time. Very often the solution will be something you'd already thought about and rejected, but just watching someone actually encounter the problem will let you see it in a whole new light. And often something else about the project has changed in the meantime (you've decided to use a different technology, for instance, or there's been a shift in your business priorities) that makes an abandoned approach suddenly feasible.
- ***Passion.*** What are the elements of the site that users really connect with? Be careful not to mistake mere enthusiasm for passion, though. You're looking for phrases like "This is exactly what I've been looking for" or "When can I start using this?"

In the course of any test, you'll also notice a number of things that are just not working like missing graphics, broken links, or typos. You should keep a list of these things so you can pass it on to whomever will fix them, but they're not what you're there to find, and you shouldn't let them distract you. Here are some things to keep in mind when you're observing:

- ***Brace yourself.*** You may be disappointed by the users' reactions. Some people just won't get it. Some just won't like it. Some will get lost and confused, apparently without reason. It can be emotionally wrenching to watch someone have a negative reaction to something you've poured your soul into. The mantra you want to have in your head is not "lt's not working!" but, rather, "What will it take to fix it?"
- ***Don't panic.*** Try to resist the temptation to jump to any conclusions until you've seen al least two users, preferably three.
- ***Be quiet.*** There's nothing more disconcerting for a test participant than the sound of laughter – or groans – coming from an adjoining room when she's having trouble using the site.
- ***Remember that you're grading on a curve.*** When a participant who uses the Internet two hours a day doesn't know how to type a URL, don't think, "Sheesh! What a dolt." Think, "How many people are there just like that out there? Can we afford to lose all of them as users?"
- ***Remember that you're seeing their best behavior.*** When you're watching a test you need to remember that people will tend to read Web pages much more thoroughly and put more effort into figuring things out in a test situation than they will in real life. After all, they're not under any time pressure, they're being paid to figure it out, and-most importantly-they don't want to look stupid. So when they can't figure something out, you have to realize that they're trying much harder than most people will and they still can't get it.
- ***Pay more attention to actions and explanations than opinions.*** Opinions expressed during user tests are notoriously unreliable. People will often exaggerate their opinions – positive and negative – because they think you want them to express strong opinions.

Reporting what you saw
As soon as possible after the test, each observer and the facilitator should type up a short list of the main problems they saw and any thoughts they have about how to fix them. You don't want to write a comprehensive report, more like an executive summary. Ideally you want the entire development team to read all of these lists (or at least scan them), so each one should be no longer than a page or two. Here's an example of the kind of notes I usually write:

USABILITY TEST NOTES: ELANCE.COM
Steve Krug
June 1, 2000

– Everybody seemed to be drawn immediately to the "Cool Stuff" links at the top of the Home page, particularly "Animate your logo: Free!" It's good that it engaged them, but after they were finished looking at the offer, they all seemed puzzled about how it related to the site.

– Two of the three users were unable to figure out what eLance was without some help. They figured it out eventually, but one of them said he wouldn't have bothered if he wasn't in a test. The wording of the main message still needs work.

– Everybody was unclear about how to get started. They all found a way, but they were uncertain and anxious along the way. There still may be too many entry points.

– Two users thought they understood the site, but they were puzzled by the Community tab. Is there another name we could use?

– The words "Buy" and "Sell" seemed to puzzle them. They weren't sure whether to think of themselves as buyers or sellers.

– Two of the three users were confused by RFP vs. Fixed-price.

- They all liked the category listing. Seeing categories they were interested in right away on the Home page encouraged them to go on and find out more.

Interpreting test results

uit: Don’t Make me Think, Steve Krug, ISBN 0-7897-2310-7

Sure, we've made mistakes.
But let's not throw the baby out with the dishes.

Lyndon Baines Johnson

OK, you've done your testing, and you've got everyone's notes. How do you decide what to change?

Review the results right away
After each round of tests, you should make time as soon as possible for the development team to review everyone's observations and decide what to do next.
At this meeting, you should distribute copies of everyone's notes and copies of whatever screens or sketches were tested. You're doing two things at this meeting:

– ***Triage:*** reviewing the problems people saw and deciding which ones need to he fixed.
– ***Problem solving:*** figuring out how to fix them.

It might seem that this would be a difficult process. After all, these are the same team members who've been arguing about the right way to do things all along. So what's going to make this session any different? Just this:

The important things that you learn from usability testing usually just make sense. They tend to be obvious to anyone who watches the sessions.

Also, the experience of seeing your handiwork through someone else's eyes will often suggest entirely new solutions for problems, or let you see an old idea in a new light.
And remember, this is acyclic process, so the team doesn't have to agree on the perfect solution. You just need to figure out what to try next.

Can this marriage be saved?
Naturally, the biggest question on everyone's mind, especially during the first few tests, is always "Are we basically on the right track?"
At these debriefing meetings, you're always trying to figure out whether the part of the site that you're testing can be made to work by tweaking it, or if you have to scrap it and take a whole different approach. For instance, if some users don't get the whole idea of the site, does it mean that the basic concept is flawed, or do you just have to change some of the wording? Here's my advice:

– ***Always consider tweaking first.*** It's rare that a site is completely off-base, but there's a tendency to be spooked by any bad user reactions, particularly after the first round of testing. Before scrapping anything, always stop and think, "What's the least we could do that might fix the problems we're seeing?" If it seems like a particular tweak has a reasonable chance of working, mock it up and test it as soon as possible.
– ***Focus on specifics.*** Try to avoid sweeping statements and focus on the precise points where people seemed to go astray. "They didn't seem to notice the navigation when they got to the second-level pages" is a much more productive place to begin problem solving than "The navigation didn't work.” One suggests that you should think about ways you could make the navigation slightly more noticeable; the other suggests you should start considering a whole new navigation scheme.
– ***Tweak, but verify.*** Remember, this is a cyclic process. Since you'll be doing another test soon, you can afford to try some tweaks before scrapping anything.

On the other hand:

– ***If the problem is deep, bite the bullet.*** I sometimes walk into situations where the site has a serious fundamental problem that nobody is talking about. The problem has usually been obvious to everyone involved for a long time, but nobody wants to be the one to mention it because (a) it seems like there's no solution, and (b) it seems like it's too late to do anything but tough it out and hope for the best. Sometimes I get called into these situations precisely because they need an outsider to announce what's already obvious to everyone.

  For instance, suppose that when you do your first test it becomes clear that the way you've chosen to organize the site isn't the way your users would organize it, and as a result they're having a very hard time figuring out where you've put things. Or perhaps you've been building your company's site assuming that your customers will want to do x and y, but when you start testing you learn that they're happy with the way they already do x and y and they're not really interested in doing them online. If you've got a deep-seated problem, a post-test debriefing meeting is a good place to finally face it. Once it's out on the table, it usually turns out to be more manageable than it seems when you're not talking about it.
– ***Remember: It's almost never too late to challenge basic assumptions.*** Rethinking the basics doesn't necessarily mean changing everything, and it often turns out that the solution to the problem is simpler than you've feared.

Typical problems
Here are the types of problems you're going to see most often when you test:

– ***Users are unclear on the concept.*** They just don't get it. They look at the site or a page and they either don't know what to make of it, or they think they do but they're wrong.
– ***The words they’re looking for aren’t there.*** This usually means that either (a) the categories you've used to organize your content aren’t the ones they would use, or (b) the categories are what they expect, but you're just not using the names they expect.
– ***There's too much going on.*** Sometimes what they're looking for is right there on the page, but they're just not seeing it. In this

case, you need to either (a) reduce the overall noise on the page, or (b) turn up the volume on the things they need to see so they "pop" out of the visual hierarchy more.

Some triage guidelines
Here's the best advice I can give you about deciding what to fix-and what not to.

- ***Ignore "kayak" problems.*** In any test, you're likely to see several cases where users will go astray momentarily but manage to get back on track almost immediately without any help. It's kind of like rolling over in a kayak, as long as the kayak rights itself quickly enough, it's all part of the so-called fun. In basketball terms, no harm, no foul.
  As long as (a) everyone who has the problem notices that they're no longer headed in the right direction quickly, and (b) they manage 10 recover without help, and (c) it doesn't seem to faze them, you can ignore the problem. In general, if the user's second guess about where to find things is always right, that's good enough. Of course, if there's an easy and obvious fix that won't break anything else, then by all means fix it. But kayak problems usually don't come as a surprise to the development team. They're usually there because of some ambiguity for which there is no simple resolution. For example, there are usually at least one or two oddball items that don't fit perfectly into any of the top-level categories of a site. So half the users may look for movie listings in Lifestyles first, and the other half will look for them in Arts first. Whatever you do, half of them are going to be wrong on their first guess, but everyone will get it on their second guess, which is fine.
- ***Resist the impulse to add things.*** When it's obvious in testing that users aren't getting something, most people's first reaction is to add something, like an explanation or some instructions.
  Very often, the right solution is to take something (or things) away that are obscuring the meaning, rather than adding yet another distraction.
- ***Take "new feature" requests with a grain of salt.*** People will often say, "I'd like it better if it could do x." It always pays to be suspicious of these requests for new features. If you probe deeper, it often turns out that they already have a perfectly fine source for x and wouldn't be likely to switch; they're just telling you what they like.
- ***Grab the low-hanging fruit.*** The main thing you're looking for in each round of testing is the big, cheap wins. These fall into two categories:
  - ***Head slappers.*** These are the surprises that show up during testing where the problem and the solution were obvious to everyone the moment they saw the first user try to muddle through. These are like found money, and you should fix them right away.
  - ***Cheap hits.*** Also try to implement any changes that (a) require almost no effort. or (b) require a little effort but are highly visible.

And finally, there's one last piece of advice about "making changes" that deserves its own section:

Don't throw the baby out with the dishes
Like any good design, successful Web pages are usually a delicate balance, and it's important to keep in mind that even a minor change can have a major impact. Sometimes the real challenge isn't fixing the problems you find-it's fixing them without breaking the parts that already work.
Whenever you're making a change, think carefully about what else is going to be affected. In particular, when you're making something more prominent than it was, consider what else might end up being de-emphasized as a result.

Eliminating Errors

uit: About Face 2.0, Alan Cooper & Robert Reinmann, ISBN 0-7645-2641-3

We discussed the bulletin dialog, issued unilaterally by a program when it is has a problem or confronts a decision that it doesn't feel capable of answering on its own. In other words, bulletin dialog boxes are used for error messages, notifiers, and confirmations, three of the most abused components of modern GUl design. With proper design, these dialogs can all but be eliminated. In this chapter, we'll explore how and why.

Errors Are Abused
There is probably no more abused idiom in the GUl world than the error dialog. The proposal that a program doesn't have the right - even the duty – to reject the user's input is so heretical that many practitioners dismiss it summarily. Yet, if we examine this assertion rationally and from the user's- rather than the programmer's - point of view, it is not only possible, but quite reasonable.

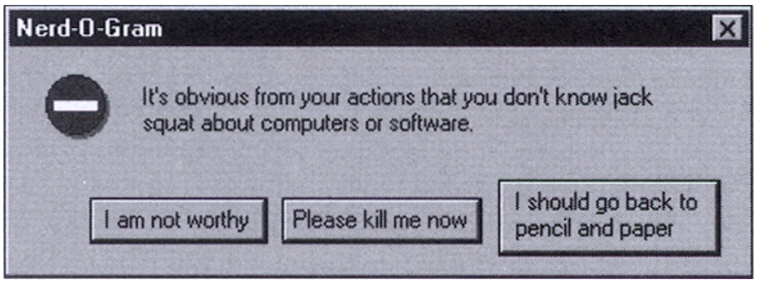
Error message boxes stop the proceedings with idiocy

Users never *want* error messages. Users want to avoid the consequences of making errors, which is very different from saying that they want error messages. It's like saying that people want to abstain from skiing when what they really want to do is avoid breaking their legs. Usability guru Donald Norman points out that users frequently blame themselves for errors in product design. Just because you aren't getting complaints from your users doesn't mean that they are happy getting error messages.

Why We Have So Many Errors
The first computers were undersized, underpowered, and expensive, and didn't lend themselves easily to software sensitivity. The operators of these machines were white-lab-coated scientists who were sympathetic to the needs of the CPU and weren't offended when handed an error message. They knew how hard the computer was working. They didn't mind getting a core dump, a bomb, an "Abort, Retry, Fail?" or the infamous "FU" message (File Unavailable). This is how the tradition of software treating people like CPUs began. Ever since the early days of computing, programmers have accepted that the proper way for software to interact with humans was to demand input and to complain when the human failed to achieve the same perfection level as the CPU.
Examples of this approach exist wherever software demands that the user do things its way instead of the software adapting to the needs of the human. Nowhere is it more prevalent, though, than in the omnipresence of error messages.

What's Wrong with Errors
Error messages, as blocking modal bulletins, must stop the proceedings with a modal dialog box. Most user interface designers - being programmers - imagine that their error message boxes are alerting the user to serious problems. This is a widespread misconception. Most error message boxes are informing the user of the inability of the program to work flexibly. Most error message boxes seem to the user like an admission of real stupidity on the program's part. In other words, to most users, error message boxes are seen not just as the program stopping the proceedings but, in clear violation of the axiom: Don't stop the proceedings with idiocy. We can significantly improve the quality of our interfaces byeliminating error message boxes.

People hate error messages
Humans have emotions and feelings: Computers don't. When one chunk of code rejects the input of another, the sending code doesn't care; it doesn't scowl, get hurt, or seek counseling. Humans, on the other hand, get angry when they are flatly told they are idiots.
When users see an error message box, it is as if another person has told them that they are stupid. Users hate this. Despite the inevitable user reaction, most programmers just shrug their shoulders and put error message boxes in anyway. They don't know how else to create reliable software.
Many programmers and user interface designers labor under the misconception that people either like or need to be told when they are wrong. This assumption is false in several ways.
The assumption that people like to know when they are wrong ignores human nature. Many people become very upset when they are informed of their mistakes and would rather not know that they did something wrong. Many people don't like to hear that they are wrong from anybody but themselves. Others are only willing to hear it from a spouse or close friend. Very few wish to hear about it from a machine. You may call it denial, but it is true, and users will blame the messenger before they blame themselves.
The assumption that users need to know when they are wrong is similarly false. How important is it for you to know that you requested an invalid type size? Most programs can make a reasonable substitution.
We consider it very impolite to tell people when they have committed some social roux pas. Telling someone they have a bit of lettuce sticking to their teeth or that their fly is open is equally embarrassing for both parties. Sensitive people look for ways to bring the problem to the attention of the victim without letting others notice. Yet programmers assume that a big, bold box in the middle of the screen that stops all the action and emits a bold "beep" is the appropriate way to behave.



It's obvious from your actions that you don't know jack squat about computers or software.

Whose mistake is it, anyway?
Conventional wisdom says that error messages tell the user when he has made some mistake. Actually, most error bulletins report to the user when the program gets confused. Users make far fewer substantive mistakes than imagined. Typical "errors" consist of the user inadvertently entering an out-of-bounds number, or entering a space where the computer doesn't allow it when the user enters something unintelligible by the computer's standards, whose fault is it? Is it the user's fault for not knowing how to use the program properly, or is it the fault of the program for not making the choices and effects clearer?
Information that is entered in an unfamiliar sequence is usually considered an error by software, but people don't have this difficulty with unfamiliar sequences.
Humans know how to wait, to bide their time until the story is complete. Software usually jumps to the erroneous conclusion that out-of-sequence input means wrong input and issues the evil error message box.
When, for example, the user creates an invoice for an invalid customer number, most programs reject the entry. They stop the proceedings with the idiocy that the user must make the customer number valid right now. Alternatively, the program could accept the transaction with the expectation that a valid customer number will eventually be entered. It could, for example, make a special notation to itself indicating what it lacks. The program then watches to make sure the user enters the necessary information to make that customer number valid before the end of the session, or even the end of the month book closing. This is the way most humans work. They don't usually enter "bad" codes. Rather, they enter codes in a sequence that the software isn't prepared to accept.
If the human forgets to fully explain things to the computer, it can, after some reasonable delay, provide more insistent signals to the user. At day's or week's end, the program can move irreconcilable transactions into a suspense account The program doesn't have to bring the proceedings to a halt with an error message. After all, the program will remember the transactions so they can be tracked down and fixed. This is the way it worked in manual systems, so why can't computerized systems do at least this much? Why stop the entire process just because something is missing? As long as the user remains well informed throughout that some accounts still need tidying, there shouldn't be a problem. The trick is to inform without stopping the proceedings.
If the program were a human assistant and it staged a sit-down strike in the middle of the accounting department because we handed it an incomplete form, we'd be pretty upset. If we were the bosses, we'd consider finding a replacement for this anal-retentive, petty, sanctimonious clerk. Just take the form, we'd say, and figure out the missing information. The authors have used Rolodex programs that demand you enter an area code with a phone number even though the person's address has already been entered. It doesn't take a lot of intelligence to make a reasonable guess at the area code. If you enter a new name with an address in Menlo Park, the program can reliably assume that their area code is 650 by looking at the other 25 people in your database who also live in Menlo Park and have 650 as their area code. Sure, if you enter a new address for, say, Boise, Idaho, the program might be stumped. But how tough is it to access a directory on the Web, or even keep a list of the 1,000 biggest cities in America along with their area codes?
Programmers may now protest: "The program might be wrong. It can't be sure. Some cities have more than one area code. It can't make that assumption without approval of the user" Not so. If we asked a human assistant to enter a client's phone contact information into our Rolodex, and neglected to mention the area code, he would accept it anyway, expecting that the area code would arrive before its absence was critical. Alternatively, he could look the address up in a directory. Let's say that the client is in Los Angeles so the directory is ambiguous: The area code could be either 213 or 310. If our human assistant rushed into the office in a panic shouting "Stop what you're doing! This client's area code is ambiguous!" we'd be sorely tempted to fire him and hire somebody with a greater-than-room-temperature IQ. Why should software be any different? A human might write 213/310? into the area code field in this case. The next time we call that client, we'll have to determine which area code is correct, but in the meantime, life can go on.
Again, squeals of protest: "But the area code field is only big enough for three digits! I can't fit 213/310? into it!" Gee, that's too bad. You mean that rendering the user interface of your program in terms of the underlying implementation model-a rigidly fixed field width-forces you to reject natural human behavior in favor of obnoxious, computer-like inflexibility supplemented with demeaning error messages? Not to put too fine a point on this, but error message boxes come from a failure of the program to behave reasonnably, not from any failure of the user.

User interface is not only skin deep

This example illustrates another important observation about user interface design. It is not only skin deep. Problems that aren't solved in the design are pushed through the system until they fall into the lap of the user. There are a variety of ways to handle the exceptional situations that arise in interaction with software-and a creative designer or programmer can probably think of a half-dozen or so off the top of her head - but most programmers just don't try. They are compromised by their schedule and their preferences, 50 they tend to envision the world in the terms of perfect CPU behavior rather than in the terms of imperfect human behavior.

Error messages don't work
There is a final irony to error messages: They don't prevent the user from making errors. We imagine that the user is staying out of trouble because our trusty error messages keep them straight, but this is a delusion. What error messages really do is prevent the program from getting into trouble. In most software, the error messages stand like sentries where the program is most sensitive, not where the user is most vulnerable, setting into concrete the idea that the program is more important than the user. Users get into plenty of trouble with our software, regardless of the quantity or quality of the error messages in it. All an error message can do is keep me from entering letters in a numeric field-it does nothing to protect me from entering the wrong numbers-which is a much more difficult design task.

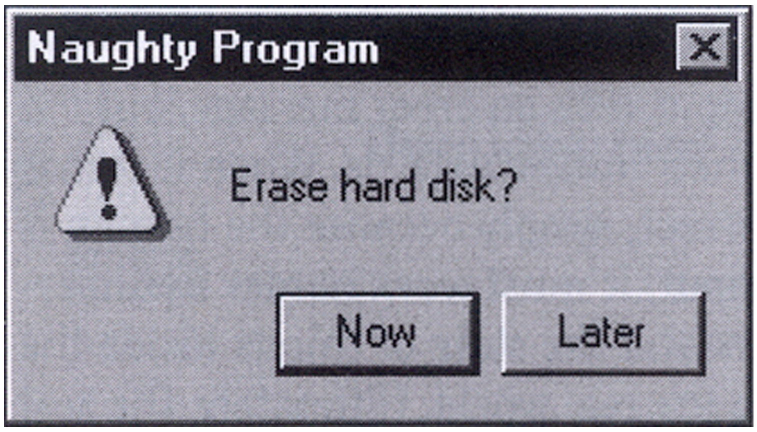
Eliminating Error Messages
We can't eliminate error messages by simply discarding the code that shows the actual error message dialog box and letting the program crash if a problem arises. Instead, we need to rewrite the programs 50 they are no longer susceptible to the problem. We must replace the error-message with a kinder, gentler, more robust software that prevents error conditions from arising, rather than having the program merely complain when things aren't going precisely the way it wants.

Like vaccinating it against a disease, we make the program immune to the problem, and then we can toss the message that reports it. To eliminate the error message, we must first eliminate the possibility of the user making the error. Instead of assuming error messages are normal, we need to think of them as abnormal solutions to rare problems-as surgery instead of aspirin. We need to treat them as an idiom of last resort.
Every good programmer knows that if module A hands invalid data to module B, module B should clearly and immediately reject the input with a suitable error indicator. Not doing this would be a great failure in the design of the interface between the modules. But human users are not modules of code. Not only should software not reject the input with an error message, but the software designer must also reevaluate the entire concept of what "invalid data" is. When it comes from a human, the software must assume that the input is correct, simply because the human is more important than the code. Instead of software rejecting input, it must work harder to understand and reconcile confusing input. The program may understand the state of things inside the computer, but only the user understands the state of things in the real world. Ultimately, the real world is more relevant and important than what the computer thinks.

Making errors impossible
Making it impossible for the user to make errors is the best way to eliminate error messages. By using bounded gizmos for all data entry, users are prevented from ever being able to enter bad numbers. Instead of forcing a user to key in his selection, present him with a list of possible selections from which to choose. Instead of making the user type in a state code, for example, let him choose from a list of valid state codes or even from a picture of a map. In other words, make it impossible for the user to enter a bad state.
Another excellent way to eliminate error messages is to make the program smart enough that it no longer needs to make unnecessary demands. Many error messages say things like "Invalid input. User must type xxxx." Why can't the program, if it knows what the user must type, just enter xxxx by itself and save the user the tongue-lashing? Instead of demanding that the user find a file on a disk, introducing the chance that the user will select the wrong file, have the program remember which files it has accessed in the past and allow a selection from that list. Another example is designing a system that gets the date from the internal clock instead of asking for input from the user.

Make errors as impossible as possible

Undoubtedly, all these solutions will cause more work for programmers. However, it is the programmer's job to satisfy the user and not vice versa. If the programmer thinks of the user as just another input device, it is easy to forget the proper pecking order in the world of software design. Users of computers aren't sympathetic to the difficulties faced by programmers. They don't see the technical rationale behind an error message box. All they see is the unwillingness of the program to deal with things in a human way. They see all error messages as some variant of the example.
One of the problems with error messages is that they are usually post facto reports of failure. They say, "Bad things just happened, and all you can do is acknowledge the catastrophe." Such reports are not helpful. And these dialog boxes always come with an OK button, requiring the user to be an accessory to the crime. These error message boxes are reminiscent of the scene in old war movies where an ill-fated soldier steps on a landmine while advancing across the rice paddy. He and his buddies clearly hear the click of the mine's triggering mechanism and the realization comes over the soldier that although he's safe now, as soon as he removes his foot from the mine, it will explode, taking some large and useful part of his body with it. Users get this feeling when they see most error message boxes, and they wish they were thousands of miles away, back in the real world.


Naughty Program
Erase hard disk?

Positive feedback
One of the reasons why software is so hard to learn is that it so rarely gives positive feedback. People learn better from positive feedback than they do from negative feedback. People want to use their software correctly and effectively, and they are motivated to learn how to make the soft- ware work for them. They don't need to be slapped on the wrist when they fail. They do need to be rewarded, or at least acknowledged, when they succeed. They will feel better about themselves if they get approval, and that good feeling will be reflected back to the product.
Advocates of negative feedback can cite numerous examples of its effectiveness in guiding people's behavior. This evidence is true, but almost universally, the context of effective punitive feedback is getting people to refrain from doing things they want to do but shouldn't: Things like not driving over 55 mph, not cheating on their spouses, and not fudging their income taxes. But when it comes to helping people do what they want to do, positive feedback is best. Imagine a hired ski instructor who yells at you, or a restaurant host who loudly announces to other patrons that your credit card was rejected.

Users get humiliated when software tells them they failed

Keep in mind that we are talking about the drawbacks of negative feedback from a computer. Negative feedback by another person, although unpleasant, can be justified in certain circumstances. One can say that the drill sergeant is at least training you in how to save your life in combat, and the imperious professor is at least preparing you for the vicissitudes of the real world. But to be given negative feedback by software - any software - is an insult. The drill sergeant and professor are at least human and have bona fide experience and merit. But to be told by software that you have failed is humiliating and degrading. Users, quite justifiably, hate to be humiliated and degraded. There is nothing that takes place inside a computer that is so important that it can justify humiliating or degrading a human user. We only resort to negative feedback out of habit.

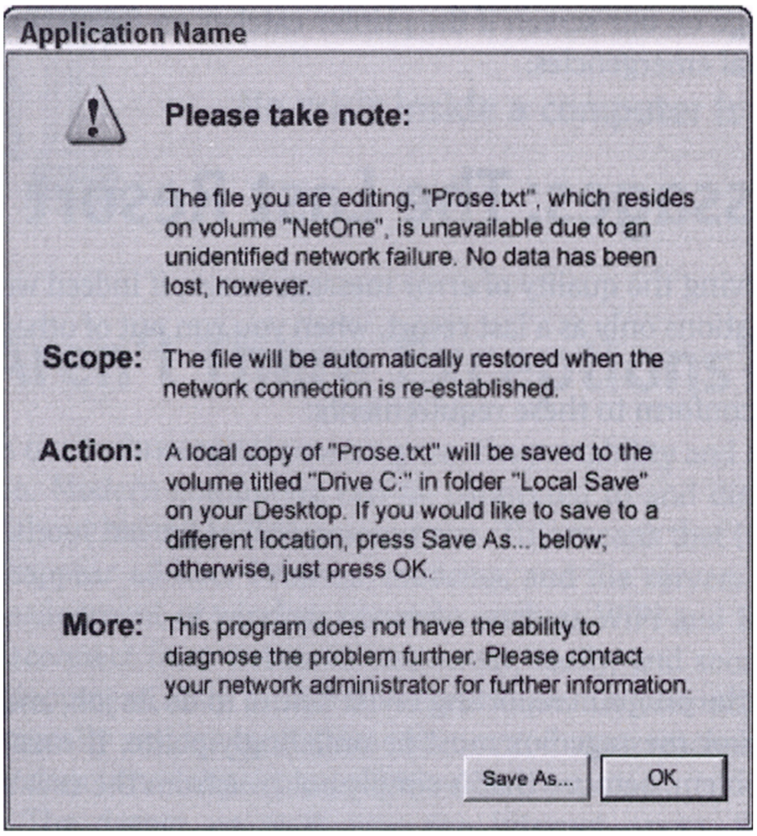
No crisis inside a computer is worth humiliating a human

Aren't There Exceptions?
As our technological powers grow, the portability and flexibility of our computer hardware grows, too. Modem computers can be connected to and disconnected from networks and peripherals without having to first power down. This means that it is now normal for hardware to appear and disappear ad hoc. Printers, modems, and file servers can come and go like the tides. With the development of wireless networks such as WiFi and Bluetooth, our computers can connect and disconnect from networks frequently, easily, and soon, transparently. Is it an error if you print a document, only to find that no printers are connected? Is it an error if the file you are editing normally resides on a drive that is no longer reachable? Is it an error if your communications modem is no longer plugged into the computer?
The deeper we wade into the Internet ocean, the more this conundrum of here-today-gone-tomorrow becomes commonplace. The Internet can easily be thought of as an infinite hard disk-one that is out of the control of any one person, company, or system administrator. A valid pointer today can be meaningless tomorrow. Is this an error?
None of these occurrences should be considered as errors. If you try to print something and there is no printer available, your program should just spool the print image to disk. The print manager program should quietly indicate when it reconnects to a printer while it has unprinted documents in its queue. This should be an aspect of normal, everyday computing. It is not an error. The same is true for files. If you open a file on the server and begin editing it, then wander out to a restaurant for lunch, taking your notebook with you, the program should see that the normal home of the file is no longer available and do something intelligent. It could use the built-in WiFi card to log onto the server remotely, or it could just save any changes you make locally, synchronizing with the version on the server when you return to the office from lunch. In any case, it is normal behavior, not an error.
Almost all error message boxes can be eliminated. If you examine the situation from the point of view that the error message box must be eliminated and that everything else is subject to change in search of this objective, you will see the truth of this assertion. You will also be surprised by how little else needs to be changed in order to achieve it. In those rare cases where the rest of the program must be altered too much, that is the time to compromise with the real world and go ahead and use an error message box. But the community of programmers needs to start thinking of this compromise as an admission of failure on the its part: that it has resorted to a low blow, a cheap shot, like a GOTO statement in its code.
Still, we would like to see an error message on our screen if the printer catches on fire. Error messages should be reserved for just such real emergencies.

Improving Error Messages: The Last Resort
Now we will discuss some methods of improving the quality of error message boxes, if indeed we are stuck using them. Use these recommendations only as a last resort, when you run out of other options. A well-formed error message box should conform to these requirements:

- Be polite
- Be illuminating
- Be helpful

Never forget that an error message box is the program reporting on its failure to do its job, and it is interrupting the


Application Name

Please take note:

The file you are editing, "Prose.txt", which resides on volume "NetOne", is unavailable due to an unidentified network failure. No data has been lost, however.

Scope: The file will be automatically restored when the network connection is re-established.

Action: A local copy of "Prose.txt" will be saved to the volume titled "Drive C:" in folder "Local Save" on your Desktop. If you would like to save to a different location, press Save As... below; otherwise, just press OK.

More: This program does not have the ability to diagnose the problem further. Please contact your network administrator for further information.

Just as there is rarely a good reason to ever use a GOTO statement in your code, there is rarely a good reason to ever issue an error message box. However, just as programmers occasionally compromise with one or two convenient GOTOs, they might occasionally issue an error message. In that case, your error message should look something like this one. It politely illuminates the problem for the user, offering him help in extracting the program from its dilemma. This error bulletin has four sections (labeled "Please take note, Scope, Action and More") that clearly help the user understand the options available and why he might choose each. The program is intelligent enough not to lose the file just because the volume became unavailable. The dialog offers an alternative action to the user by way of the Save As... button.

user to do this. The error message box must be unfailingly polite. It must never even hint that the user caused this problem, because that is simply not true from the user's perspective. The customer is always right.
The user may indeed have entered some goofy data, but the program is in no position to argue and blame. It should do its best to deliver to the user what he asked for, no matter how silly. Above all, the program must not, when the user finally discovers his silliness, say, in effect, "Weil, you did something really stupid, and now you can't recover. Too bad." It is the program's responsibility to protect the user even when he takes inappropriate action. This may seem draconian, but it certainly isn't the user's responsibility to protect the computer from taking inappropriate action.
The error message box must illuminate the problem for the user. This means that it must give him the kind of information he needs to make an appropriate determination to solve the program's problem. It needs to make clear the scope of the problem, what the alternatives are, what the program will do as a default, and what information was lost, if any. The program should treat this as a confession, telling the user everything.
It is wrong, however, for the program to just dump the problem on the user's lap and wipe its hands of the matter. It should directly offer to implement at least one suggested solution right there on the error message box. It should offer buttons that will take care of the problem in various ways. If a printer is missing, the message box should offer options for deferring the printout or selecting another printer. If the database is hopelessly trashed and useless, it should offer to rebuild it to a working state, including telling the user how long that process will take and what side effects it will cause.

Above an example of a reasonable error message. Notice that it is polite, illuminating, and helpful. It doesn't even hint that the user's behavior is anything but impeccable.

The End of Errors
Error message boxes validate the idea that the computer is the final arbiter of correctness, and the user is there just to serve its digital majesty. This attitude influences both programmers and users. It tempts programmers to make bad judgments in design and to take shortcuts in implementation. These compromises necessitate the use of yet more error messages. Also, users are anesthetized by error messages so they cannot visualize the possibility of error-free computing.

Tijd versus functionaliteit – MoSCoW

uit: DSDM, Jennifer Stapleton, ISBN 90-395-109101

Leven met tijdslimieten wil niet zeggen dat er harder gewerkt moet worden of dat er meer uren gedraaid moeten worden dan normaliter het geval is. Natuurlijk is er de druk om erg hard te werken en naarmate het project vordert, zullen de ontwikkelaars af en toe veel uren maken om de deadline te halen. Dit kan echter bij ieder project voorkomen, onafhankelijk van de gebruikte methode. Lange werkdagen mogen voor DSDM-ontwikkelaars niet als standaard gelden; het is de opzet dat men de normale kantooruren aanhoudt en de avonden en weekends vrij heeft. Om dit te bereiken moet op alle niveaus de wijze veranderen waarop het werk wordt gestuurd; ten aanzien van het project zelf, het team en het individu.

4.1 IJzer met handen breken
Op het niveau van het project moet de lat niet hoger liggen dan haalbaar is. Het is erg belangrijk dat hierover tijdens de Bedrijfsanalyse overeenstemming wordt bereikt. Dit vormt de basis voor alle besluiten die in de beschikbare tijd genomen kunnen worden. Het komt vaak voor dat in latere fasen van het project - wanneer de gebruikers een beter idee krijgen van het systeem - wordt gevraagd om bepaalde zaken toe te voegen. Mocht dit gebeuren, dan is het niet de taak van de projectmanager om deze wijziging door te voeren en te gaan onderhandelen over meer mensen en tijd. De doorlooptijd is bij her begin vastgesteld en het inhuren van meer personeel kan de aflevering op de afgesproken datum in gevaar brengen. Brooks maakte hier al melding van, maar toch zoekt men hierin vaak de oplossing. Hij merkte toen op dat als er nieuwe medewerkers aan het project gaan deelnemen, veel tijd wordt gestoken in het bijpraten van de nieuwkomers. Dit gaat ten koste van het werk dat de oorspronkelijke teamleden hadden moeten doen.

Om deze reden is het af te raden meer personeel of meer tijd te verlangen. Het enige wat men echt kan doen, is een ander gepland onderdeel weglaten, aangenomen dat het nieuwe verzoek absoluut noodzakelijk is. Toch is er dan een probleem, want het weg te laten onderdeel droeg bij aan de afgesproken functionaliteit en de kwaliteit van het geheel mag hier beslist niet onder lijden. In paragraaf 4.3 komt aan de orde hoe dit met de MoSCoW-regels aangepakt kan worden.

Op het niveau van het team gelden de prioriteiten van de functionele en niet-functionele eisen om te beslissen wat het team moet gaan doen en wanneer. Als een zelfstandige groep mensen kunnen zij zelf besluiten wie welke werkzaamheden op welk moment doet. De teamleden werken samen om aan de eisen met de hoogste prioriteit te voldoen. Het regelmatig en veelvuldig houden van besprekingen is essentieel om de ontwikkeling op het juiste spoor te houden. Dagelijkse bijeenkomsten van een half uur lijken in het begin een aanslag op de 'nuttig' te besteden tijd, maar juist daarbij nen veel zaken snel geregeld worden en het bevordert het denken als team in plaats van als individu. De dagelijkse besprekingen zijn een formele vervanging van het praatje bij de koffieautomaat, waar vaak de goede ideeën en deskundige kennis van anderen worden overgenomen. Als een team kiest voor een wekelijkse bijeenkomst, kan het gebeuren dat het werk dat iemand de afgelopen week heeft verricht in strijd is met het werk van iemand anders; en dan moet het dus herzien worden. Het is niet juist om ervan uit te gaan dat mensen met elkaar praten omdat ze in één team zitten.

Het ergste voorbeeld van niet-communiceren (en daarom niet-samenwerken) bij een team kwam ik tegen bij een project waar de vier analisten binnen één team werkten aan vier aaneengeschoven bureaus zonder tussenschot. Ze zagen elkaar dus de hele dag, maar ze praatten niet! Drie van hen hielden zich bezig met procesanalyse en de vierde met gegevensanalyse. Toen ik daar was, waren ze al drie maanden aan het werk en het project was al behoorlijk uitgelopen. Geen van de procesanalisten had met de anderen over de interfaces van hun onderdeel gesproken en de gegevensanalist werkte volkomen onafhankelijk van de rest van het team. Her was dan ook niet vetwonderlijk dat zij vaak op overlappende terreinen aan het werk waren en dat er al veel energie was gestoken in het oplossen van gemeenschappelijke problemen. Een wekelijkse bijeenkomst zou voor deze onfortuinlijke groep zeker waardevol geweest zijn. Nu werkte iedereen alleen voor zichzelf aan de taken die de projectmanager had opgedragen.

Op het niveau van het individu moeten ontwikkelaars en gebruikers accepteren dat het niet mogelijk is om alles te doen. Ontwikkelaars die ieder technisch detail van het geautomatiseerde systeem willen onderzoeken vanuit hun vakkennis en die niet in staat zijn die dingen te doen die voor her bedrijf werkelijk van belang zijn, vinden DSDM een vermoeiende manier van werken. Bovendien verstoren zij de werkzaamheden van de andere leden van het team. De anderen zijn wel gericht op de gebruikers en accepteren dat genoeg doen uiteindelijk de beste aanpak betekent. Helaas bestaat er geen methode waarmee men van tevoren kan vaststellen of DSDM iemand zal bevallen - dat blijkt pas tijdens de rit. Sommige mensen zijn direct overtuigd, maar anderen kunnen daar wel een maand voor nodig hebben. Ik werkte ooit aan een project waarvan de voortgang ernstig belemmerd werd door het optreden van een ontwikkellaar. Deze had op het gebied van programmeren een uitstekende staat van dienst en hij kon heel snel nieuwe technieken leren, maar hij kon het gewoon niet verkroppen dat hij af en toe werd beperkt in wat hij belangrijk vond. Nog erger was dat hij snel geïrriteerd raakte door gebruikers die wel eens van mening veranderden. Dat verhinderde hem immers zijn ontwerp-, bouw- en testwerkzaamheden uit te voeren!

4.2 Timeboxes
Er zijn verschillende definities in omloop van het begrip timebox. Vaak gaat het om de tijd tussen het begin en het einde van een project. De einddatum ligt vast en het systeem dient op die datum te worden afgeleverd. Bij DSDM is het idee van timeboxes verder doorgevoerd doordat ze zijn ingebed in het totale project. Hiermee wordt een reeks vaste deadlines verkregen voor het leveren van iets, waarbij 'iets' een analysemodel (geheel of gedeeltelijk) kan zijn, een deel van een front-end, een compleet gebied

van functionaliteit, een combinatie van dit soort dingen, of wat ook maar te bedenken valt om het project dichter bij het leveren van een goed product op een vaste einddatum te brengen.

Bij DSDM variëren timeboxes in het algemeen van twee tot zes weken lang - hoe korter hoe beter. Deze omvang is echter niet heilig, hoewel er wel eens mensen zijn die zeven weken niet acceptabel vinden. Als de projectmanager en het team zeven weken als minimum zien voor een bepaalde hoeveelheid werk, is dat op zich geen probleem. Alleen is het grote voordeel van korte timeboxes dat men eenvoudiger kan inschatten wat in die tijd kan worden gedaan.

Het kost negen maanden om een baby te krijgen. Tijdens het wachten op de blijde gebeurtenis besteden de ouders veel tijd aan het uitwerken van kleertjes, een wieg, iets aan de muur, knuffels, enzovoort. Als zij echter te horen krijgen dat de baby al over een week komt, moeten de ouders alles binnen een week aanschaffen en zorgen dat de echt essentiële zaken op tijd in huis zijn. Dankzij deze manier van denken wordt het gebruik van timeboxes tot een effectief middel bij projecten met een korte tijdsduur. Het is eenvoudiger je voor te stellen wat in korte tijd kan worden gedaan, dan te schatten hoeveel tijd een pas verkregen opdracht gaat vergen. Vandaar dat de timebox een nuttig middel is bij het schatten van de menskracht die nodig is voor het maken van het systeem.

Bij timeboxes is het van belang dat ze niet op activiteiten zijn gebaseerd. Het doel van een timebox is iets te maken. Hoe dat gebeurt, wordt besloten door de mensen die het werk doen. Dit is een manier waarop het derde principe van DSDM over het regelmatig opleveren van producten wordt toegepast.

4.3 De MoSCoW-regels

In de eerste versies van het DSDM-handboek waren de MoSCoW-regels niet te vinden, maar bij veel organisaties zijn ze in het kader van DSDM ingevoerd; het blijkt een uitstekende manier te zijn om bij een project de onderlinge prioriteiten van eisen te beheren. Ze zijn bedacht door Dai Clegg van Oracle, een van de eerste deelnemers in het Consortium. *MoSCoW* is een acroniem voor de prioritering die aan de eisen wordt toegekend. De twee o's hebben geen betekenis, maar de hoofdletters staan voor:

- ***Must-have*** - voor eisen die essentieel voor het systeem zijn. Als deze ontbreken, is het systeem onbruikbaar en waardeloos. De Must-haves geven aan wat bij DSDM de 'minimaal bruikbare deelverzameling' (minimum usable subset) wordt genoemd.

- ***Should-have*** - voor belangrijke eisen die wellicht ook als essentieel zouden zijn beschouwd als de beschikbare tijd het toegelaten had. Zonodig kan het systeem er echter buiten.

- ***Could-have*** - betreft eisen die wat eenvoudiger weggelaten kunnen worden bij een deeloplevering van het systeem.

- ***Want to have but will not have this time round*** – voor die waardevolle eisen die wel kunnen wachten totdat er verder wordt ontwikkeld.

Al deze eisen zijn noodzakelijk voor het volledige systeem. Het wensenlijstje is hierin overigens niet opgenomen. Het belangrijke punt van MoSCoW is het leveren van een stevige basis voor beslissingen over de werkzaamheden gedurende het totale project en ook binnen iedere timebox.

Als praktisch voorbeeld van het gebruik van MoSCoW nemen we het maken van een videorecorder. De Must-haves kunnen zijn: het opnemen van programma's, het terugspoelen van banden en het afspelen ervan. Zonder deze eigenschappen kan het apparaat eenvoudigweg niet werken. Bij Should-have kan het gaan om de mogelijkheid op te nemen zonder dat men thuis is. Hierdoor wordt de waarde voor de gebruiker zonder meer vergroot, maar het apparaat kan ook zonder deze mogelijkheid. De Could-haves kunnen dan zaken zijn als snel vooruitspoelen en stilzetten. Hiermee wordt het gebruiksgemak vergroot, maar deze zaken zijn niet essentieel. De want-to-have zou afstandsbediening, tracking, enzovoort kunnen zijn: voorzieningen die men ook later nog kan toevoegen.

The Medium is Messy

uit: Eye - Winter 1998, Andy Cameron

Interactivity demands a new sense of engagement. Time to stop designing earnestly and start playing ... seriously. The traditional ideal of the design process seems a long way from my experiences with interactive media. The word "design" evokes the image of purposeful creative labour, solving a well-defined problem in an efficient and elegant manner. We find it difficult to work in this way at the Antirom studio in London. Here the creative process is often unstructured and messy, with only the vaguest idea al the start as to where the whole thing might end up. It feels like playing with things rather than designing them. And in most cases there is no preconception of what the end product of an experiment might turn out to he, simply the hope that we will recognize something good if we stumble across it.

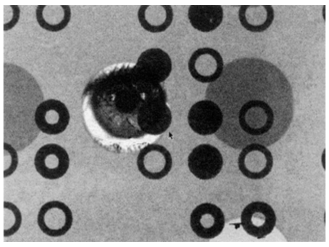
Within a context like this, it is hard not to think like a Modernist about interactive media. It feels as if interactivity has an essence waiting to be revealed - some approaches seem intrinsically more suited to the nature of the medium than others. It is easy enough to reproduce a traditional linear form, slap buttons on to it and call it "interactive story", "interactive book", "interactive cinema" or what have you. But such hybrids rarely

DESIGNERS MAY FIND THE IDEA FRIVOLOUS. GAMES ARE OFTEN CONSIDERED EPHEMERAL AND RATHER SILLY.

have anything to do with the real power of interactive media. What is the power of interactivity anyway? Borges suggests that it is an intrinsically dull mode of expression when he describes its characteristics as "symmetry, arbitrary rules, medium"
I believe that the crucial formal aspects of interactive media - how they operate as a language, what forms and figures of rhetoric they make available and (most crucially) what kind of spectatorship they offer - are hardly understood. At Antirom we try to make interactive media that work on their own terms and succeed in holding the audience's attention. We set out to play with a new medium. To find out what you can do with video, text or holography on a computer screen, or what happens when you connect an image with a sound via the movement of the cursor. Our experiments are wide-ranging and open-ended and we have come to terms with our ignorance about the way in which interactive media work. We try just about everything we can

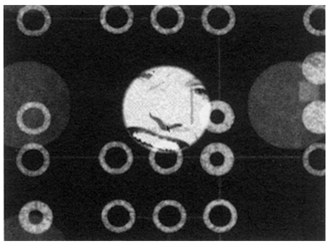
IT'S LIKE PLAYING WITH THINGS RATHER THAN DESIGNING. WE HOPE WE WILL RECOGNISE SOMETHING GOOD IF WE STUMBLE ACROSS IT.

think of - and that our rudimentary programming skills allow us to construct. We have not discarded lines of enquiry merely because they appeared to he ridiculous or stupid,



and we have made as many mistakes as we could, as quickly as we could. Most importantly, we have found that designing for a medium so new that it bas not yet evolved a cultural form; or a language of its own, is almost impossible. The only way to approach it is to play, to explore it creatively and without preconceptions. And in a curious and unexpected symmetry between method and content, we find that the pieces we like best (and which work best with audiences) are also playful. In fact it appears that a quality of playfulness is essential to all engaging interactive representations: play and interactivity are interconnected at a deep level.

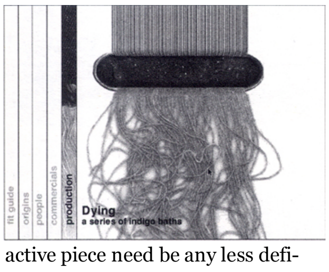
Why should this be? One approach to thinking about this is to consider the different ways in which time can be represented, and in particular the relationship between what might be called the time of the representation itself, and the time of its reading. In traditional (i.e. non-interactive) media, information is presented in the form of enunciation, a "speech" act which takes place before the audience receives the message. So the time of speaking comes before the time of listening. The time of writing before the time of reading. As spectators we are positioned outside the time of the representation itself. A linear



text exercises an authority that is dispersed by interactivity. An interactive representation is information in waiting, it refers to a principle, a set of rules. an algorithm, a status outside time, that can simulate events and information in time. The

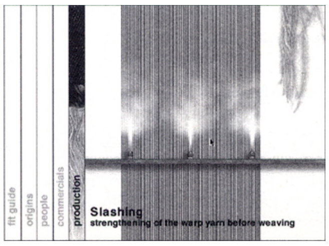
THE ILLUSION OF INTERACTIVE MEDIA IS TO PERSUADE US THAT IT ONLY "SPEAKS" WHEN WE ARE ACTIVELY ENGAGED WITH IT

referent, the thing other than itself to which the simulation refers, is the condition for communication, not the message itself. The creation of a message - the cutting out and sequencing of information from the mass of data - is effected by the spectator, albeit within a framework of conditions designed by the author. The central illusion of interactive media is to persuade us that we are inside the time of the representation that the representation only speaks when we are actively engaged with it This means that rather than being passive receivers of a message uttered earlier, we are active participants within a situation, now. Within such a situation, information is communicated in real time, at the very moment and only at that moment when we explore, experiment and play with it. The information appears to be triggered by the audience in that situation, rather than being some external thing consumed by them. So there is, displacement of the author's (or designer's) "voice" - a situation does not "speak" to us in the way that, text does. When we discover information from within the situation, the voice appears to be our own. This would suggest that the object of the design process in interactive media might be considered more in terms of, model rather than, proposition. The design "product" is a representation within which the audience explores and discovers information for themselves, rather than being given it directly in the form of, message. This does not imply that the information carried within an inter-



active piece need be any less definite or precise than that carried by more traditional form - the message of, model can be as clear and as unambiguous as an imperative statement - but it should emerge from what appears to be an open-ended and playful engagement with the information on the part of the audience.

Of course, this approach is not appropriate for all information design. My CD-ROM of the Encyclopedia Brittanica is not playful in its design - it is what the computer games industry calls a "port", a transformation of a work from one platform to another - in this case from a series of leather-bound volumes to a CD-ROM. The interface is designed la make the retrieval of information as simple and as fast as possible. Here, interactivity is being used as an indexing tool rather than as a fully developed language of its own. A problem with the notion of playfulness is the cultural baggage it brings in tow. Enthusiasts argue that interactive media are superior to traditional media because of the apparent empowerment they offer the audience, and the concomitant lessening of the power and control of the authorial voice. But paradoxically an interactive message, by concealing its own voice, may be capable of a more powerful manipulation of its audience. A model or a game can be set up to create whatever meaning the designers intend, but this may not be apparent to the audience. By hiding its own voice, an interactive representation can conceal its bias, and present itself as matter of fact This problem is as much about the novelty of the medium as anything else. As audiences become acclimatized they may become better equipped to deconstruct such products. I think it is too early to draw more than tentative



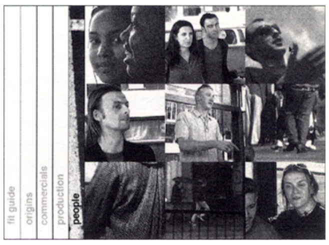
and general conclusions about how this radical new form of representation will affect the ways in which we communicate and the ways in

which we design for communication. But it seems clear that information will be structured more and more as a matrix of "what if" scenarios, which outline the ways in which actions are linked together in multi-dimensional chains of cause and effect. Though ideas of "information space" will be developed (3D spatial interactive metaphors are the first clichés of the new medium - think of Navigator; or Explorer. or "where do you want to go today" or "superhighway": better still, think of another metaphor), a situation is so much more than just a space. Some designers may find the idea of making playful design somewhat frivolous. Games, after all, are not considered to be part of serious culture, but are ephemeral and rather silly. Many college computer rooms have notices on the wall declaring that game-playing is prohibited. The game is not "work" but a diversion from work, nor is it a proper object of serious study. The game is something which, although tolerated, the law must seek to repress, to keep in its proper place.

AN ACTIVE MESSAGE MAY BE CAPABLE OF A MORE POWERFUL MANIPULATION OF ITS AUDIENCE

Yet in the class I teach in interactive media at the University of Westminster, I encourage the students to play computer games. This gives them a sense of the possibilities - and limitations- of the most highly developed form of interactive media any of us have experienced: the video games developed by Sony, Nintendo, Microsoft, EA and smaller British developers such as Psygnosis, Eidos, Rare and Bullfrog.

This playing around did not initially meet with the approval of the department. Yet while few people accuse film students of "just wanting to watch films" there is still a linge-



ring suspicion that those students who take the module in interactivity just want to play video games. Playfulness has the same relation to interactive media that "scopophilia" [the pleasure of looking] has to cinema. It is the basis upon which everything else is built. Designing playful engagement into interactive media will become an increasingly serious game in the future. It demands a fundamentally different engagement from the audience- and the designer- and the design community had better get used to the idea.

Usability Slogans

uit: Usability Enigineering, Jakob Nielson, ISBN 0-12-518406-9

Major parts of the usability approach in this book can be summarized in the short slogans given here. You will find that some of the slogans contradict each other. Unfortunately, usability is filled with apparent contradictions that are only resolved after more detailed analysis. Some contradictions and trade-offs will always remain, and it is the job of the usability engineer to arrive at the best solution for the individual project's needs. There are very few hard and firm rules in usability that do not have some exceptions. For the full story, read on.

Your Best Guess Is Not Good Enough

A basic reason for the existence of usability engineering is that it is impossible to design an optimal user interface just by giving it your best try. Users have infinite potential for making unexpected misinterpretations of interface elements and for performing their job in a different way than you imagine.

Your design will be much better if you work on the basis of an understanding of the users and their tasks. Then, by all means design the best interface you can, but make sure to validate it with user tests and the other methods recommended in this book. It is no shame to have to revise a user interface design as a result of user testing. This happens to the best of usability experts, and it might indeed be a true measure of usability maturity that one is willing to acknowledge the need to modify initial design choices to accommodate the users. Julius Caesar is widely acknowledged as one of the greatest generals of antiquity. Even so, his true talent was not perfect campaign planning but his ability to adjust to the situation as it evolved. He of ten placed his legions in highly problematic situations, which they only survived because he changed his plans to accommodate the facts. If Caesar could conquer France by admitting his mistakes, then maybe you can win some market share by admitting yours.

The User Is Always Right

As mentioned, all experience shows that any initial attempt at a user interface design will include some usability problems. Therefore, the user interface developer needs to acquire a certain design humility and acknowledge the need to modify the original design to accommodate the user's problems. The designer's attitude should be that if users have problems with an aspect of the interface, then this is not because they are stupid or just should have tried a little harder. Somebody once tested the usability of a user manual and found that users almost always made a mistake in a certain step of a particular procedure. Their solution was to frame the difficult step in a box and add a note saying "Read these instructions carefully!" Of course, the correct conclusion would have been that the description was too difficult and should be rewritten.

The User Is Not Always Right

Unfortunately, it does not follow that user interface designs can be derived just by asking users what they would like. Users of ten do not know what is good for them. One example is a study of the weight of telephone handsets conducted in the 1950s when people were used to fairly heavy handsets. The result of asking users whether they would like lighter handsets was no, they were happy with the handsets they had. Even 50, a test of handsets that looked identical but had different weights showed that people preferred handsets with about half the then normal weight.

Users have a very hard time predicting how they will interact with potential future systems with which they have no experience. As another example, 73% of the respondents in a survey of 9,652 commuters said that they would not use a proposed information service with continual up-to-the-minute traffic information. But after they were shown sample screens from a prototype of the service, 84% of the respondents said that they would in fact use it.

Furthermore, users will often have divergent opinions when asked about details of user interface design. For example, studies of how people name things have shown that the probability of having two people apply the same name to an object is between 7% and 18%, depending on the object, clearly making it infeasible to design command names just by asking some user.

Users Are Not Designers

The ideal solution to the usability question might be to leave the design of the interface up to the individual users. Just provide sufficient customization flexibility, and all users can have exactly the interface they like. Studies have shown, however, that novice users do not customize their interfaces even when such facilities are available. One novice user exclaimed, "I didn't dare touch them [the customization features] in case something went wrong." Therefore, a good initial interface is needed to support novice users. Expert users (especially programmers) do use customization features, but there are still compelling reasons not to rely on user customization as the main element of user interface design. First, customization is easy only if it builds on a coherent design with good previously designed options from which to choose. Second, the customization feature itself will need a user interface and will thus add to the complexity of the system and to the users' learning load. Third, too much customization leads each user to have a wildly different interface from the interfaces used by other users. Such interface variety makes it difficult to get help from colleagues, even though that is the help method rated highest by both novice and expert users. And fourth, users may not always make the most appropriate design decisions. For example, Grudin and Barnard compared command abbreviations they defined with abbreviations defined by individual users, and found that users made about twice as many errors when using their own abbreviations. Even when given the chance to redefine their abbreviations after the experiment, six of seven test users kept their poor abbreviation sets virtually intact, typically explaining that while yes, they had some problems with it, it seemed as good as any other set they could think of. Of course, users have other jobs and do not work as user interface professionals.

Designers Are Not Users

System designers are human and they certainly use computers: both characteristics of users. Therefore, it can be tempting for designers to trust their own intuition about user interface

issues, since they do share these two important characteristics of the real users. Unfortunately, system designers are different from users in several respects, including their general computer experience (and enthusiasm) and their knowledge of the conceptual foundation of the design of the system. When you have a deep understanding of the structure of a system, it is normally easy to fit a small extra piece of information into the picture and interpret it correctly. Consequently, a system designer may look at any given screen design or error message and believe that it makes perfect sense, even though the same screen or message would be completely incomprehensible to a user who did not have the same under- standing of the system.
Knowing about a system is a one-way street. One cannot go back to knowing nothing. It is almost impossible to disregard the information one already knows when trying to assess whether another piece of information would be easy to understand for a novice user.
Landauer uses a hidden animal picture as an analogy for developers' understanding of their own system. Hidden animal games, as well as popular children's books of the type, "Where is so-and-so?" show images with various levels of details, among which is the animal or character one is supposed to find. Initially, it is very difficult to pick the animal out of the background, but once you have seen where the animal is, it is very easy to see it again. In fact, it is impossible to ignore one's knowledge of where the animal is and regain a perspective on the picture where one would have to search to find the animal.
A survey of 2,000 adults in Oregon showed that only 18% could use a bus schedule to find the time of departure. This finding does not indicate that the remaining 82% of Oregonians are less intelligent and should never be allowed on a bus. Instead, the likely explanation is that the bus schedule was designed by people with extensive knowledge of buses and local transportation who just knew the meaning of every element on the schedule, and therefore never considered that parts of it might be difficult to understand for people who rarely take a bus.

Vice Presidents Are Not Users
Many CEOs and other top corporate executives have started to realize that usability is becoming one of their main competitive parameters, as user interfaces account for a steadily higher proportion of the value added in their products and services. The downside of this higher visibility for user interfaces is that these executives may start meddling in user interface design.
Vice presidents and other corporate executives should realize that they are no more representative of the end users than the developers are. With the possible exception of management information systems and other software intended for vice presidents, corporate executives in a high-tech company are very different from the average user, and their intuitions about what would make a great design may not be accurate.
Apple reports that they sometimes had “a powerful person” in their company propose changes to their interface. They avoided making these changes by pointing out that this person probably had very different characteristics than the intended users and that their design had been tested on such real users. Of course, all design suggestions should be welcomed in order to serve as inspiration, but one should never be unduly swayed by a comment from a single person. People get promoted to vice president because of their managerial and decision-making skills, not because of their design skills.

Less is More
One tempting solution to the user interface design problem might be to throw in any imaginable option or feature. If everything is there, then everybody should be satisfied, right?
Wrong. Every single element in a user interface places some additional burden on the user in terms of having to consider whether to use that element. Having fewer options will of ten mean better usability because the users can then concentrate on understanding those fewer options. Software reviewers are becoming aware that more features are not always better, and a major popular computer magazine ran a cover story lamenting the tendency of some programs to double in size every two years, coining the term "fatware" to describe bloated software.

Details Matter
Unfortunately, usability of ten depends on minor interface details, which is why systematic usability engineering work is necessary to ferret out those details. For example, Unilever reports on the development of instructions for a frozen-dinner microwave indicator that would gradually change from being white to being blue. User testing showed that the phrase "turns blue" was much poorer than "white disappears" for describing this change, even though the two phrases are logically equivalent relative to this process. The blue color was not uniform – it was dark blue in some places and light blue in others – so users were uncertain "how blue is blue?" when the first wording was used.

Help Doesn't
Sometimes, online help and documentation doesn't really help the users. That is to say, users often do not find the information they want in the mass of possible help and documentation and, even if they do find it, they may misinterpret the help. Also, help adds an extra set of features to a system, thus complicating the interface just by virtue of existing. In any case, the possibility for providing help should not been seen as an excuse to design a needlessly complex interface. It is always better if users can operate the system without having to refer to a help system. Usability is not a quality that can be spread out to cover a poor design like a thick layer of peanut butler, so a user-hostile interface does not get user-friendly even by the addition of a brilliant help system.

Usability Engineering Is Process
Most of this book consists of advice for activities to perform as part of the system development process. Readers may sometimes lose patience and wish that I had just told them about the result rather than the process: What makes an interface good? Unfortunately, so many things sometimes make an interface good and sometimes make it bad that any detailed advice regarding the end product has to be embellished with caveats, to an extent that makes it close to useless, not least because there will often be several conflicting guidelines. In contrast, the usability engineering process is well established and applies equally to all user interface designs. Each project is different, and each final user interface will look different, but the activities needed to arrive at a good result are fairly constant.

Thinking Aloud

uit: Usability Enigineering, Jakob Nielson, ISBN 0-12-518406-9

Thinking aloud may be the single most valuable usability engineering method. Basically, a thinking-aloud test involves having a test subject use the system while continuously thinking out loud. By verbalizing their thoughts, the test users enable us to understand how they view the computer system, and this again makes it easy to identify the users' major misconceptions. One gets a very direct understanding of what parts of the dialogue cause the most problems, because the thinking-aloud method shows how users interpret each individual interface item.

The thinking-aloud method has traditionally been used as a psychological research method, but it is increasingly being used for the practical evaluation of human- computer interfaces. The main disadvantage of the method is that is does not lend itself very well to most types of performance measurement. On the contrary, its strength is the wealth of qualitative data it can collect from a fairly small number of users. Also, the users' comments often contain vivid and explicit quotes that can be used to make the test report more readable and memorable.

At the same time, thinking aloud may also give a false impression of the cause of usability problems if too much weight is given to the users' own "theories" of what caused trouble and what would help. For example, users may be observed to overlook a certain field in a dialog box during the first part of a test. After they finally find the field, they may claim that they would have seen it immediately if it had been in some other part of the dialog box. It is important not to rely on such statements. Instead, the experimenter should make notes of what the users were doing during the part of the experiment where they overlooked the critical field. Data showing where users actually looked has much higher validity than the users' claim that they would have seen the field if it had been somewhere else. The strength of the thinking-aloud method is to show what the users are doing and why they are doing it while they are doing it in order to avoid later rationalizations.

Thinking out loud seems very unnatural to most people, and some test users have great difficulties in keeping up a steady stream of utterances as they use a system. Not only can the unnaturalness of the thinking aloud situation make the test harder to conduct, but it can also impact the results. First, the need to verbalize can slow users down, thus making any performance measurements less representative of the users' regular working speed. Second, users' problem solving behavior can be influenced by the very fact that they are verbalizing their thoughts. The users might notice inconsistencies in their own models of the system, or they may concentrate more on critical task components, and these changes may cause them to learn some user interfaces faster or differently than they otherwise would have done. For example, Berry and Broadbent provided users with written instructions on how to perform a certain task and found that users performed 9% faster if they were asked to think aloud while doing the task. Berry and Broadbent argue that the verbalization reinforced those aspects of the instructions, which the users needed for the task, thus helping them become more efficient. In another study, users who were thinking aloud while performing various file system operations were found to make only about 20% of the errors made by users who were working silently. Furthermore, the users in the thinking-aloud study finished their tasks about twice as fast as the users in the silent condition.

The experimenter will often need to continuously prompt the user to think out loud by asking questions like, "What are you thinking now?" and "What do you think this message means?" (after the user has noticed the message and is clearly spending time looking at it and thinking about it). If the user asks a question like, "Can I do such-and-such?" the experimenter should not answer, but instead keep the user talking with a counter-question like, "What do you think will happen if you do it?" If the user acts surprised after a system action but does not otherwise say anything, the experimenter may prompt the user with a question like, " that what you expected would happen?" Of course, following the general principle of not interfering in the user's use of the system, the experimenter should not use prompts like, "What do you think the message on the bottom means?" if the user has not noticed that message yet.

Since thinking aloud seems strange to many people, it may help to give the test users a role model by letting them observe a short thinking-aloud test before the start of their own experiment. One possibility is for the experimenter to enact a small test, where the experimenter performs some everyday task like looking up a term in a dictionary while thinking out loud. Alternatively, users can be shown a short video of a test that was made with the sole purpose of instructing users. Showing users how a test videotape looks may also help alleviate their own fears of any videotaping that will be done during the test.

Users will often make comments regarding aspects of the user interface, which they like or do not like. To some extent, it is one of the great advantages of the thinking-aloud method that one can collect such informal comments about small irritants that would not show up in other forms of testing. They may not impact measurable usability, but they might as well be fixed. Unfortunately, users will often disagree about such irritants, so one should take care not to change an interface just because of a comment by a single user. Also, user comments will often be inappropriate when seen in a larger interface design perspective, so it is the responsibility of the experimenter to interpret the user's comments and not just accept them indiscriminately. For example, users who are using a mouse for the first time will of ten direct a large proportion of their comments toward aspects of moving the mouse and pointing and clicking, which might be interesting for a designer of more intuitive input hardware but are of limited use to a software designer. In such a test, the experimenter would need to abstract from the users' mouse problems and try to identify the underlying usability problems in the dialogue and estimate how the users would have used the interface if they had been better at using the pointing device.

Constructive Interaction
A variation of the thinking-aloud method is called constructive interaction and involves having two test users use a system together. This method is sometimes also called codiscovery learning. The main advantage of constructive interaction is that the test situation is much more natural than standard thinking-aloud tests with single users, since people are used to verbalizing when they are trying to solve a problem together. Therefore, users may make more comments when engaged in constructive interaction than when simply thinking aloud for the benefit of an experimenter. The method does have the disadvantage that the users may have different strategies for learning and using computers. Therefore, the test session may switch back and forth between disparate ways of using the interface, and one may also occasionally find that the two test users simply cannot work together.

Constructive interaction is especially suited for usability testing of user interfaces for children since it may be difficult to get them to follow the instructions for a standard thinking-aloud test.

Constructive interaction is most suited for projects where it is easy to get large numbers of users into the lab, and where these users are comparatively cheap, since it requires the use of twice as many test users as single-user thinking aloud.

Retrospective Testing
If a videotape has been made of a user test session, it becomes possible to collect additional information by having the user review the recording. This method is sometimes called retrospective testing. The user's comments while reviewing the tape are sometimes more extensive than comments made under the (at least perceived) duress of working on the test task, and it is of course possible for the experimenter to stop the tape and question the user in more detail without fearing to interfere with the test, which has essentially already been completed. Retrospective testing is especially valuable in cases where representative test users are difficult to get hold of, since it becomes possible to gain more information from each test user. The obvious downside is that each test takes at least two times as long, so the method is not suited if the users are highly paid or perform critical work from which they cannot be spa red for long. Unfortunately, those users who are difficult to get to participate in user testing are of ten exactly those who are also very expensive, but there are still some cases where retrospective testing is beneficial.

Coaching Method
The coaching method is somewhat different from other usability test methods in having an explicit interaction between the test subject and the experimenter (or "coach"). In most other methods, the experimenter tries to interfere as little as possible with the subject's use of the computer, but the coaching method actually involves steering the user in the right direction while using the system.

During a coaching study, the test user is allowed to ask any system-related question of an expert coach who will answer to the best of his or her ability. Usually, the experimenter or a research assistant serves as the coach. One variant of the method involves a separate coach chosen from a population of expert users. Having an independent coach lets the experimenter study how the coach answers the user's questions. This variant can be used to analyze the expert coach's model of the interface. Normally, though, coaching focuses on the novice user and is aimed at discovering the information needs of such users in order to provide better training and documentation, as well as possibly redesigning the interface to avoid the need for the questions.

The coaching method has proven helpful in getting Japanese users to externalize their problems while using computers. Other, more traditional methods are sometimes difficult to use in Japan, where cultural norms make some people reluctant to verbalize disagreement with an interface design.

The coaching situation is more natural than the thinking-aloud situation. It also has an advantage in cases where test users are hard to come by because the intended user population is small, specialized, and highly paid. Coaching provides the test users with tangible benefits in return for participating in the test by giving them instruction on a one-to-one basis by a highly skilled coach.

Finally, the coaching method may be used in cases where one wants to conduct tests with expert users without having any experts available. Coaching can bring novice users up to speed fairly rapidly and can then be followed by more traditional tests of the users' performance once they have reached the desired level of expertise.

Designing Pages for Scanning not Reading

uit: Don't Make me Think, Steve Krug, ISBN 0-7897-2310-7

If you | Don't know | Whose signs | These are
You can't have | Driven very far | Burma-Shave

Sequence of billboards promoting shaving cream, circa 1935

Faced with the fact that your users are whizzing by, there are five important things you can do to make sure they see-and understand-as much of your site as possible:

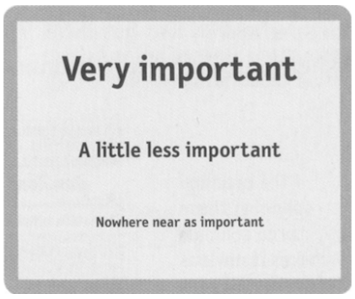
- Create a clear visual hierarchy on each page
- Take advantage of conventions
- Break pages up into clearly defined areas
- Make it obvious what's clickable
- Minimize noise.

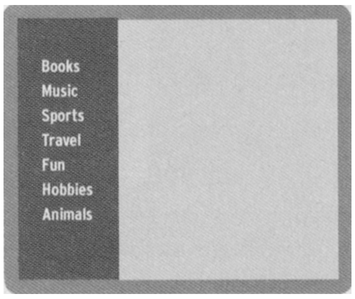
Create a clear visual hierarchy
One of the best ways to make a page easy to grasp in a hurry is to make sure that the appearance of the things on the page-all of the visual cues-clearly and accurately portray the relationships between the things on the page: which things are related, and which things are part of other things. In other words, each page should have a clear visual hierarchy. Pages with a clear visual hierarchy have three traits:

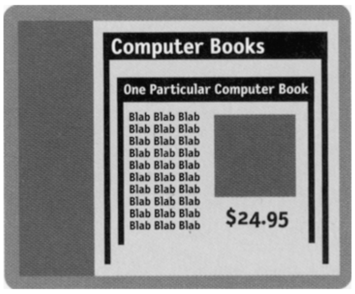
- ***The more important something is, the more prominent it is.*** For instance, the most important headings are either larger, bolder, in a distinctive color, set off by more white space, or nearer the top of the page-or some combination of the above.
- ***Things that are related logically are also related visually.*** For instance, you can show that things are similar by grouping them together under a heading, displaying them in a similar visual style, or putting them all in a clearly defined area.
- ***Things are "nested" visually to show what's part of what.*** For instance, a section heading ("Computer Books") would appear above the title of a particular book, visually encompassing the whole content area of the page, because the book is part of the section. And the title in turn would span the elements that describe the book.

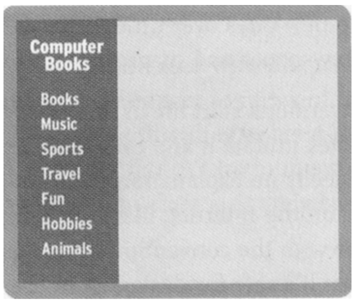
There's nothing new about visual hierarchies. Every newspaper page, for instance, uses prominence, grouping, and nesting to give us useful information about the contents of the page before we read a word. This picture goes with this story because they're both spanned by this headline. This story is the most important because it has the biggest headline, the widest column, and a prominent position on the page.
We all parse visual hierarchies – online and on paper-every day, but it happens so quickly that the only time we're even vaguely aware that we're doing it is when we can't do it-when the visual cues (or absence of them) force us to think. A good visual hierarchy saves us work by preprocessing the page for us, organizing and prioritizing its contents in a way that we can grasp almost instantly.
But when a page doesn't have a clear visual hierarchy-if everything looks equally important, for instance-we're reduced to the much slower process of scanning the page for revealing words and phrases, and then trying to form our own sense of what's important and how things are organized. It's a lot more work.
Besides, we want editorial guidance in Web sites, the same way we want it in other media. The publisher knows better than anyone which pieces of the site's content are most important, valuable, or popular, so why not identify them for me and save me the trouble?
Parsing a page with a visual hierarchy that's even slightly flawed-where a heading spans things that aren't part of it, for instance-is like reading a carelessly constri.1cted sentence ("Bill put the cat on the table for a minute because it was a little wobbly."). Even though we can usually figure out what the sentence is supposed to mean, it still throws us momentarily and forces us to think when we shouldn't have to.

Conventions are your friends
At some point in our youth, without ever being taught, we all learned to read a newspaper. Not the words, but the conventions. We learned, for instance, that a phrase in very large





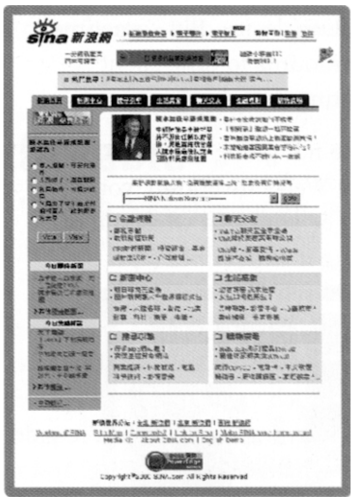




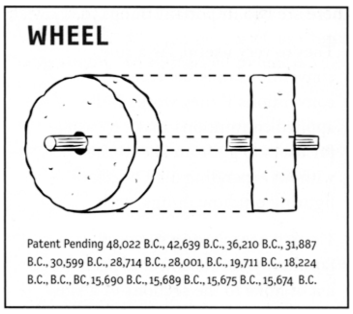
This flawed visual hierarchy suggests that all of the sections of the site are part of the Computer books section.

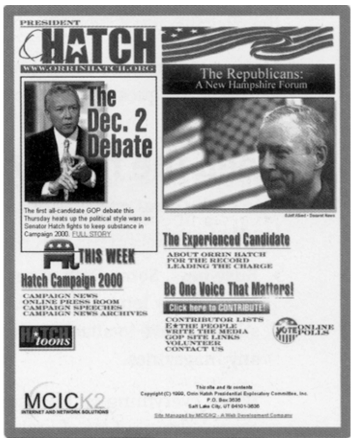
San Jose Mercury News
Phone bills to fall in summer
Coastal oil spill danger reduced
Gonzales pushes slate of planners

The headline spanning three columns makes it obvious that they're all part of the same story. The size of the big headline makes it clear at a glance that this is the most important story.



Conventions enable users to figure out a lot about a Web page, even if they can't understand a word of it.


WHEEL
Patent Pending 48,022 B.C., 42,639 B.C., 36,210 B.C., 31,887 B.C., 30,599 B.C., 28,714 B.C., 28,001, B.C., 19,711 B.C., 18,224 B.C., B.C., BC, 15,690 B.C., 15,689 B.C., 15,675 B.C., 15,674 B.C.


PRESIDENT
HATCH
The Dec. 2 Debate
The Republicans: A New Hampshire Forum
The Experienced Candidate
THIS WEEK
Hatch Campaign 2000
Be One Voice That Matters!
Click here to CONTRIBUTE!
MCICK2

type is usually a headline that summarizes the story underneath it, and that text underneath a picture is either a caption that tells me what it's a picture of, or-if it's in very small type-a photo credit that tells me who took the picture.

We learned that knowing the various conventions of page layout and formatting made it easier and faster to scan a newspaper and find the stories we were interested in. And when we started traveling to other cities, we learned that all newspapers used the same conventions (with slight variations), so knowing the conventions made it easy to read any newspaper. Every publishing medium develops conventions and continues to refine them and develop new ones over time. The Web already has a lot of them, mostly derived from newspaper and magazine conventions, and new ones will continue to appear.

All conventions start life as somebody's bright idea. If the idea works well enough, other sites imitate it and eventually enough people have seen it in enough places that it needs no explanation. This adoption process takes time, but it happens pretty quickly on the Internet, like everything else. For instance, enough people are now familiar with the convention of using a metaphorical shopping cart on e-commerce sites that it's safe for designers to use a shopping cart icon without labeling it "Shopping cart".

There are two important things to know about Web conventions:

– ***They're very useful.*** As a rule, conventions only become conventions if they work. Well-applied conventions make it easier for users to go from site to site without expending a lot of effort figuring out how things work.
  There's a reassuring sense of familiarity, for instance, in seeing a list of links to the sections of a site on a colored background down the left side of the page, even if it's sometimes accompanied by a tedious sense of déjà vu.
– ***Designers are often reluctant to take advantage of them.*** Faced with the prospect of using a convention, there's a great temptation for designers to reinvent the wheel instead, largely because they feel (not incorrectly) that they've been hired to do something new and different, and not the same old thing. (Not to mention the fact that praise from peers, awards, and high-profile job offers are rarely based on criteria like "best use of conventions.")
  Sometimes time spent reinventing the wheel results in a revolutionary new rolling device. But sometimes it just amounts to time spent reinventing the wheel. If you're not going to use an existing Web convention, you need to be sure that what you're replacing it with either (a) is so clear and self-explanatory that there's no learning curve-so it's as good as a convention, or (b) adds so much value that it's worth a small learning curve. If you're going to innovate, you have to understand the value of what you're replacing, and many designers tend to underestimate just how much value conventions provide.

My recommendation: Innovate when you know you have a better idea (and everyone you show it to says "Wow!"), but take advantage of conventions when you don't.

Break up pages into dearly defined areas

Ideally, users should be able to play a version of Dick Clark's old game show $25,000 Pyramid with any well-designed Web page. Glancing around, they should be able to point at the different areas of the page and say, "Things I can do on this site!" "Links to today's top stories!" "Products this company sells!" "Things they're eager to sell me!" "Navigation to get to the rest of the site!" Dividing the page into clearly defined areas is important because it allows users to decide quickly which areas of the page to focus on and which areas they can safely ignore. Several of the initial eye-tracking studies of Web page scanning suggest that users decide very quickly which parts of the page are likely to have useful information and then almost never look at the other parts-almost as though they weren't there.

Make it obvious what's clickable

Since a large part of what people are doing on the Web is looking for the next thing to click, it's important to make it obvious what's clickable and what's not.

For example, on Senator Orrin Hatch's Home page during his unsuccessful 2000 presidential bid, it wasn't clear whether everything was click- able, or nothing was. There were 18 links on the page, but only two of them invited you to click by their appearance: a large button labeled "Click here to CONTRIBUTE!" and an underlined text link ("FULL STORY"). The rest of the links were colored text. But the problem was that all of the text on the

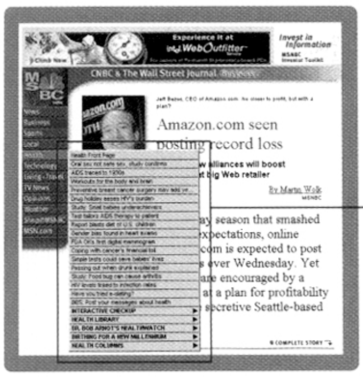
page was in color, so there was no way to distinguish the links at a glance.
It's not a disastrous flaw. I'm sure it didn't take most users long to just start clicking on things. But when you force users to think about something that should be mindless like what's clickable, you're squandering the limited reservoir of patience and goodwill that each user brings to a new site.
One of my other favorite examples is the search box at drkoop.com (C. Everett Koop's health site).
Every time I use it, it makes me think, because the button that executes the search just doesn't look like a button-in spite of the fact that it has two terrific visual cues: It contains the word "SEARCH," which is one of the two perfect labels for a search box button, and it's the only thing near the search box. It even has a little triangular arrow graphic, which is one of the Web's conventional "Click here" indicators. But the arrow is pointing away from the text, as though it's pointing at something else, while the convention calls for it to be pointing toward the clickable text.
Moving the arrow to the left would be enough to get rid of the question mark over my head.

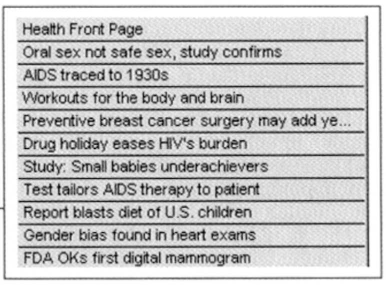
Keep the noise down to a dull roar
One of the great enemies of easy-to-grasp pages is visual noise. There are really two kinds of noise:

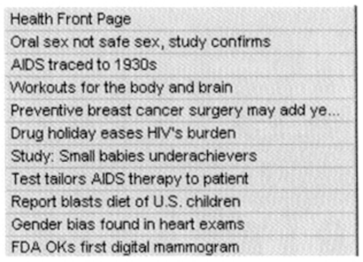
– ***Busy-ness.*** Some Web pages give me the same feeling I get when I'm wading through my letter from Publisher's Clearing House trying to figure out which sticker I have to attach to the form to enter without accidentally subscribing to any magazines.
  When everything on the page is clamoring for my attention the effect can be overwhelming: Lots of invitations to buy! Lots of exclamation points and bright colors! A lot of shouting going on!)
– ***Background noise.*** Some pages are like being at a cocktail party; no one source of noise is loud enough to be distracting by itself, but there are a lot of tiny bits of visual noise that wear us down.

For instance, MSNBC's menus are a powerful and slick navigation device that let users get to any story in the site quickly. But the lines between items add a lot of noise. Graying the lines would make the menus much easier to scan.

Users have varying tolerances for complexity and distractions; some people have no problem with busy pages and background noise, but many do. When you're designing Web pages, it's probably a good idea to assume that everything is visual noise until proven otherwise.




Health Front Page
Oral sex not safe sex, study confirms
AIDS traced to 1930s
Workouts for the body and brain
Preventive breast cancer surgery may add ye...
Drug holiday eases HIV's burden
Study: Small babies underachievers
Test tailors AIDS therapy to patient
Report blasts diet of U.S. children
Gender bias found in heart exams
FDA OKs first digital mammogram
Before


Health Front Page
Oral sex not safe sex, study confirms
AIDS traced to 1930s
Workouts for the body and brain
Preventive breast cancer surgery may add ye...
Drug holiday eases HIV's burden
Study: Small babies underachievers
Test tailors AIDS therapy to patient
Report blasts diet of U.S. children
Gender bias found in heart exams
FDA OKs first digital mammogram
After

Why Users Like Mindless Choices

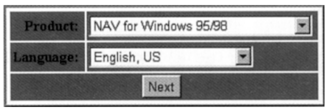
uit: Don’t Make me Think, Steve Krug, ISBN 0-7897-2310-7

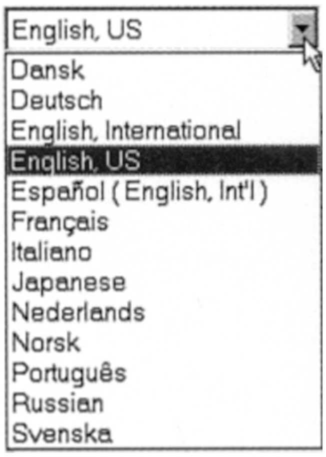
It doesn't matter how many times I have to click, as long as each click is a mindless, unambiguous choice.

Krug's second law of usability

Web designers and usability professionals have spent a lot of time over the years debating how many times you can expect users to click to get what they want without getting too frustrated. Some sites even have design rules stating that it should never take more than a specified number of clicks (usually three, four, or live) to get to any page in the site.

On the face of it, "number of clicks to get anywhere" seems like a useful criteria. But over time I've come to think that what really counts is not the number of clicks it takes me to get to what I want (although there are limits), but rather how hard each click is-the amount of thought required, and the amount of uncertainty about whether I'm making the right choice.
In general, I think it's safe to say that users don't mind a lot of clicks as long as each click is painless and they have continued confidence that they're on the right track. I think the rule of thumb might be something like "three mindless, unambiguous clicks equal one click that requires thought."
The classic first question in the word game Twenty Questions - "Animal, vegetable, or mineral?" – is a wonderful example of a mindless choice. As long as you accept the premise that anything that's not a plant or an animal-including things as diverse as pianos, limericks, and encyclopedias, for instance-falls under "mineral," it requires no thought at all to answer the question correctly. Unfortunately, most choices on the Web aren't as clear.


Product: NAV for Windows 95/98
Language: English, US
Next


English, US
Dansk
Deutsch
English, International
English, US
Español (English, Int'l)
Français
Italiano
Japanese
Nederlands
Norsk
Português
Russian
Svenska

For instance, if I go to Symantec's Virus Updates page because I want to update my copy of Norton AntiVirus, I'm faced with two choices l have to make before l can continue. One of the choices, Language, is relatively painless. It takes only a tiny bit of thought for me to conclude that "English, US" means "United States English," as opposed to "English, UK."
If I bothered to click on the pull-down menu, though, I'd realize that I was actually just muddling through, since there is no "English, UK" on the list. I'd also probably be a little puzzled by "Espanol (English, Int'l)" but I wouldn't lose any sleep over it.
The other choice, Product, is a bit dicier, however.
The problem is that it refers to "NAV for Windows 95/98." Now, I'm sure that it's perfectly clear to everyone who works at Symantec that NA V and "Norton AntiVirus" are the same, but it requires at least a small leap of faith on my part. And even though I know for certain that I'm using Windows 98, there's at least the tiniest question in my mind whether that's exactly the same as "Windows 95/98." Maybe there is something called "Windows 95/98" that I just don't know about.

Another example: When I'm trying to buy a product or service to use in my home office. I often encounter sites that ask me to make a choice like...

- Home
- Office

Which one is me? It's the same way I feel when I'm standing in front of two mail boxes labeled Stamped Mail and Metered Mail with a business reply card in my hand. What do they think it is-stamped or metered? And what happens if I drop it in the wrong box? The point is, we face choices all the time on the Web and making the choices mindless is one of the main things that make a site easy to use.

Omit ~~Needless~~ Words

uit: Don’t Make me Think, Steve Krug, ISBN 0-7897-2310-7

Get rid of half the words on each page,
then get rid of half of what's left.

Krug's third law of usability

Of the five or six things that I learned in college, the one that has stuck with me the longest-and benefited me the most-is E. B. White's seventeenth rule in The Elements of Style:

17. Omit needless words. Vigorous writing is concise. A sentence should contain no unnecessary words, a paragraph no unnecessary sentences, for the same reason that a drawing should have no unnecessary lines and a machine no unnecessary parts.

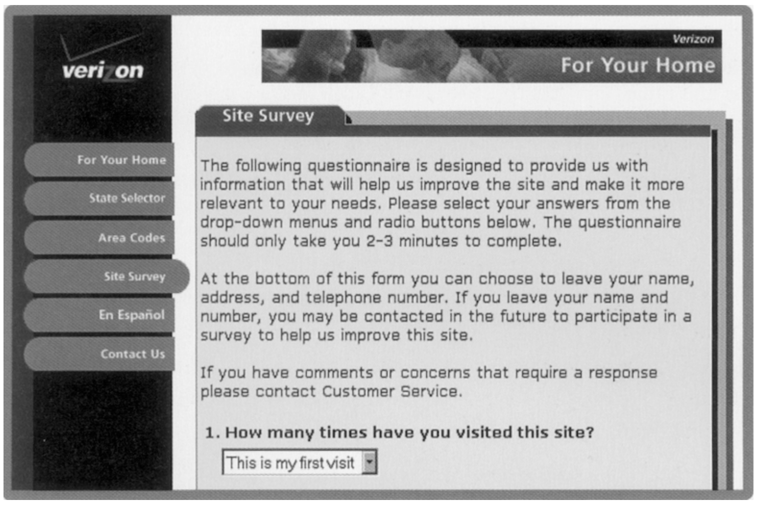
When I look at most Web pages, I'm struck by the fact that most of the words I see are just taking up space, because no one is ever going to read them. And just by being there, all the extra words suggest that you may actually need to read them to understand what's going on, which often makes pages seem more daunting than they actually are. My Third Law probably sounds excessive, because it's meant to. Removing half of the words is actually a realistic goal; I find I have no trouble getting rid of half the words on most Web pages without losing anything of value. But the idea of removing half of what's left is just my way of trying to encourage people to be ruthless about it. Getting rid of all those words that no one is going to read has several beneficial effects:

- It reduces the noise level of the page.
- It makes the useful content more prominent.
- It makes the pages shorter, allowing users to see more of each page at a glance without scrolling.

I’m not suggesting that the articles at Salon.com should be shorter than they are. I'm really talking about two specific kinds of writing: happy talk and instructions.

Happy talk must die
We all know happy talk when we see it: it's the introductory text that's supposed to welcome us to the site and tell us how great it is, or to tell us what we're about to see in the section we've just entered,
If you're not sure whether something is happy talk, there's one sure-fire test: if you listen very closely while you're reading it, you can actually hear a tiny voice in the back of your head saying, "Blah blah blah blah blah..... A lot of happy talk is the kind of self-congratulatory promotional writing that you find in badly written brochures. Unlike good promotional copy, it conveys no useful information, and it focuses on saying how great we are, as opposed to delineating what makes us great.
Although happy talk is sometimes found on Home pages-usually in paragraphs that start with the words "Welcome to....-its favored habitat is the front pages of the sections of a site (“section fronts"). Since these pages are often just a table of contents with no real content of their own, there's a temptation to fill them with happy talk. Unfortunately, the effect is as if a book publisher felt obligated to add a paragraph to the table of contents page saying, "This book contains many interesting chapters about -, -, and -' We hope you enjoy them. Happy talk is like small talk-content free, basically just a way to be sociable. But most Web users don't have time for small talk: they want to get right to the beef. You can-and should-eliminate as much happy talk as possible.

Instructions must die
The other major source of needless words is instructions. The main thing you need to know about instructions is that no one is going to read them-at least not until after repeated attempts at "muddling through" have failed. And even then, if the instructions are wordy, the odds of users finding the information they need is pretty low. Your objective should always be to eliminate instructions entirely by making everything self-explanatory, or as close to it as possible. When instructions are absolutely necessary, cut them back to the bare minimum.



The following questionnaire is designed to provide us with information that will help us improve the site and make it more relevant to your needs. Please select your answers from the drop-down menus and radio buttons below. The questionnaire should only take you 2-3 minutes to complete.

At the bottom of this form you can choose to leave your name, address, and telephone number. If you leave your name and number, you may be contacted in the future to participate in a survey to help us improve this site.

If you have comments or concerns that require a response please contact Customer Service.

1. How many times have you visited this site?
This is my first visit

For example, when I click on Site Survey at the Verizon site, I get an entire screen full of instructions to read.
I think same aggressive pruning makes them much more useful:

BEFORE: 103 WORDS	
The following questionnaire is designed to provide us with information that will help us improve the site and make it more relevant to your needs.	The first sentence is just introductory happy talk. I know what a survey is for; all l need is the words "help us" to show me that they understand that I'm doing them a favour by filling it out.
Please select your answers from the drop-down menus and radio buttons below.	Most users don't need to be told how to fill in a Web form, and the ones who do won't know what a "drop-down menu" and a "radio button" are anyway.
The questionnaire should only take you 2-3 minutes to complete.	At this point, I'm still trying to decide whether to bother with this questionnaire, so knowing that it's short is useful information.
At the bottom of this form you can choose to leave your name, address, and telephone number. If you leave your name and number, you may be contacted in the future to participate in a survey to help us improve this site.	This instruction is of no use to me at this point. It belongs at the end of the questionnaire where I can act on it. As it is, its only effect is to make the instructions look daunting.
If you have comments or concerns that require a response I please contact Customer Service.	The fact that I shouldn't use this form if I want an answer is useful and important information. Unfortunately, though, they don't bother telling me how I contact Customer Service-or better still, giving me a link so I can do it from right here.

AFTER: 41 WORDS
Please help us improve the site by answering these questions. It should only take you 2-3 minutes to complete this survey. Note: If you have comments or concerns that require a response don’t use this form. Instead, please contact Customer Service.

Consistency and Standards

uit: About Face 2.0, Alan Cooper & Robert Reinmann, ISBN 0-7645-2641-3

Many in-house usability organizations view themselves, among other things, as the gatekeepers of consistency in digital product design. Consistency implies a similar look, feel, and behaviour across the various modules of a software product, and this is sometimes extended to apply across all the products a vendor sells. For at-large software vendors, such as Macromedia and Adobe, who regularly acquire new software titles from smaller vendors, the branding concerns of consistency take on a particular urgency. It is obviously in their best interests to make acquired software look as though it belongs, as a first-class offering, alongside products developed in-house. Beyond this, both Apple and Microsoft have an interest in encouraging their own and third-party developers to create applications that have the look and feel of the OS platform on which the program is being run, so that the user perceives their respective platforms as providing a seamless and comfortable user experience.

Benefits of interface standards
User interface standards provide benefits that address these issues, although they come at a price. Standards provide benefits to users when executed appropriately. According to Jakob Nielsen, relying on a single interface standard improves users' ability to quickly learn interfaces and enhances their productivity by raising throughput and reducing errors. These benefits accrue because users are more readily able to predict program behaviour based on past experience with other parts of the interface, or with other applications following similar standards.
At the same time, interface standards also benefit software vendors. Customer training and technical support costs are reduced because the consistency that standards bring improves ease of use and learning. Development time and effort are also reduced because formal interface standards provide ready-made decisions on the rendering of the interface that development teams would otherwise be forced to debate during project meetings. Finally, good standards can lead to reduced maintenance costs and improved reuse of design and code.

Risks of interface standards
The primary risk of any standard is that the product that follows it is only as good as the standard itself. Great care must be made in developing the standard in the first place to make sure, as Nielsen says, that the standard specifies a truly usable interface, and that it is usable by the developers who must build the interface according to its specifications.
It is also risky to see interface standards as a panacea for good interfaces. Most interface standards emphasize the syntax of the interface, its visual look and feel, but say little about deeper behaviours of the interface or about its higher-level logical and organizational structure. There is a good reason for this: A general interface standard has no knowledge of context incorporated into its formalizations. It takes into account no specific user behaviours and usage patterns within a context, but rather focuses on general issues of human perception and cognition and, sometimes, visual branding as well. These concerns are important, but they are presentation details, not the interaction framework upon which such rules hang.

Standards, guidelines, and rules of thumb
Although standards are unarguably useful, they need to evolve as technology and our understanding of users and their goals evolve. Some practitioners and programmers invoke Apple's or Microsoft's user interface standards as if they were delivered from Mt. Sinai on a tablet. Both companies publish user interface standards, but both companies also freely and frequently violate them and update the guidelines post facto. When Microsoft proposes an interface standard, it has no qualms about changing it for something better in the next version. This is only natural- interface design is still in its infancy, and it is wrongheaded to think that there is benefit in standards that stifle true innovation. In some respects, Apple's dramatic visual shift from OS 9 to OS X has helped to dispel the notion among the Mac faithful that interface standards are etched in granite.
The original Macintosh was a spectacular achievement precisely because it transcended all Apple's previous platforms and standards. Conversely, much of the strength of the Mac came from the fact that vendors followed Apple's lead and made their interfaces look, work, and act alike. Similarly, many successful Windows programs are unabashedly mode led after Word, Excel, and Outlook.
Interface standards are thus most appropriately treated as detailed guidelines or rules of thumb. Following interface guidelines too rigidly or without careful consideration of the needs of users in context can result in force-fitting an application's interface into an inappropriate interaction model.

When to violate guidelines
So, what should we make of interface guidelines? Instead of asking if we should follow standards, it is more useful to ask: When should we violate standards? The answer is when, and only when, we have a very good reason.

Obey standards unless there is a truly superior alternative

But what constitutes a very good reason? Is it when a new idiom is measurably better? Usually, this sort of measurement can be quite elusive because it rarely reduces to a quantifiable factor alone. The best answer is: When an idiom is clearly seen to be significantly better by most people in the target user audience (your personas) who try it, there's a good reason to keep it in the interface. This is how the toolbar came into existence, along with outline views, tabs, and many other idioms. Researchers may have been examining these artefacts in the lab, but it was their useful presence in real-world software that confirmed the success.

Your reasons for diverging from guidelines may ultimately not prove to be good enough and your product may suffer. But you and other designers will learn from the mistake. This is what Christopher Alexander calls the "unselfconscious process," an indigenous and unexamined process of slow and tiny forward increments as individuals attempt to improve solutions. New idioms (as well as new uses for old idioms) pose a risk, which is why careful, goal-directed design and appropriate testing with real users in real working conditions are so important.

Consistency and standards across applications
Using standards or guidelines has special challenges when a company that sells multiple software titles decides that all its various products must be completely consistent from a user-interface perspective.
From the perspective of visual branding, as discussed earlier, this makes a great deal of sense, although there are some intricacies. If an analysis of personas and markets indicates that there is little overlap between the users of two distinct products and that their goals and needs are also quite distinct, you might question whether it makes more sense to develop two visual brands that speak specifically to these different customers, rather than using a single, less-targeted look. When it comes to the behaviour of the software, these issues become even more urgent. A single standard might be important if customers will be using the products together as a suite. But even in this case, should a graphics-oriented presentation application, like PowerPoint, share an inter- face structure with a text processor like Word? Microsoft's intentions were good, but it went a little too far enforcing global style guides. PowerPoint doesn't gain much from having a similar menu structure to Excel and Word, and it loses quite a bit in ease-of-use by conforming to an alien structure that diverges from the user's mental models. On the other hand, the designers did draw the line somewhere, and PowerPoint does have a slide-sorter display, an interface unique to that application.

Consistency doesn't imply rigidity

Designers, then, should bear in mind that consistency doesn't imply rigidity, especially where it isn't appropriate. Interface and interaction style guidelines need to grow and evolve like the software they help describe. Sometimes you must bend the rules to best serve your users and their goals (and sometimes even your company's goals). When this has to happen, try to make changes and additions that are compatible with standards. The spirit of the law, not the letter of the law, should be your guide.

Labeling Systems

Uit: Information Architecture, Louis Rosenfeld & Peter Morville, ISBN 0-596-00035-9

Labeling is a form of representation. Just as we use spoken words to represent concepts and thoughts, we use labels to represent larger chunks of information in our web sites. For example, "Contact Us" is a label that represents a chunk of information, often including a contact name, an address, and telephone, fax, and email information. You cannot present all this information quickly and effectively on an already crowded web page without overwhelming impatient users who might not actually need that information. Instead, a label like "Contact Us" works as a shortcut that triggers the right association in the user's mind without presenting all that stuff prominently. The user can then decide whether to click through or read on to get more contact information. So the goal of a label is to communicate information efficiently; that is, without taking up too much of a page's vertical space or a user's cognitive space.

Unlike the weather, hardly anyone ever talks about labeling (aside from a few deranged librarians, linguists, journalists, and, increasingly, information architects), but everyone can do something about it. In fact, we are doing something about it, albeit unconsciously: anyone developing content or an architecture for a web site is creating labels without even realizing it. And our label creation goes far beyond our web sites; ever since Adam named the animals, labeling has been one of the things that make us human. Spoken language is essentially a labeling system for concepts and things. Perhaps because we constantly label, we take the act of labeling for granted. That's why the labeling on web sites is often poor, and users suffer the consequences. This chapter provides some advice on how to think through a site's labeling before diving into implementation.

How does labeling fit with the other systems we've discussed? Well, labels are often the most obvious way of clearly showing the user your organization and navigation systems. For example, a single web page might contain different groups of labels, with each group representing a different organization or navigation system. Examples include labels that match the site's organization system (e.g., Home/Home Office, Small Business, Medium & Large Business, Government, Health Care), a side-wide navigation system (e.g., Main, Search, Feedback), and a subsite navigation system (e.g., Add to Cart, Enter Billing Information, Confirm Purchase).

Why You Should Care About Labeling

Prerecorded or canned communications, including prim, the Web, scripted radio, and TV, are very different from interactive real-time communications. When we talk with another person, we rely on constant user feedback to help us hone the way we get our message across. We subconsciously notice our conversation partner zoning out, getting ready to make their own point, or beginning to clench their fingers into an angry fist, and we react by shifting our own style of communication, perhaps by raising our speaking volume, increasing our use of body language, changing a rhetorical tack, or fleeing.

Unfortunately, when we "converse" with users through the web sites we design, the feedback isn't quite so immediate, if it exists at all. In fact, a site serves as an intermediary that translates messages in content form from the site's owners and authors to users and back again. This "telephone game" muddies the message. So in such a dis-intermediated medium with no visual or other cues, communicating is harder, and labeling is therefore more important.

To minimize this disconnect, information architects must try their best to design labels that speak the same language as a site's users while reflecting its content. And, just as in a dialogue, when there is a question or confusion over a label, there should be clarification and explanation. Labels should educate users about new concepts. The conversation between user and site owner generally begins on a site's main page. To get a sense of how successful this conversation might be, take a site's main page and ask yourself a few questions: Do the prominent labels on this page stand out to you? If they do, why? (Often, successful labels are invisible; they don't get in your way.) If a label is new, unanticipated, or confusing, is there an explanation? Or are you required to click through to learn more? Although unscientific, this label testing exercise will help you get a sense of how the conversation might go with actual users. Let's try it with an average, run-of-the-mill main page from the U-Haul site.

The U-Haul main page's labels don't seem terribly out of the ordinary. However, mediocrity isn't an indicator of value or success; in fact, many trouble spots arise from an informal cruise through the page's labels. We've identified them as follows:

Main - "Main" refers to what? In web parlance, "Main" typically has something to do with a main page. Here it describes a set of useful link labels such as "Get Rates & Reservations" and "Find a U-Haul Location." Why label these important links as "Main"? There are other possible labels, or visual design techniques could have been used to make them stand out without mixing things up with a conventional term like "Main." What exactly will be found under "College Connection"? It sounds like a branded program; while it may represent useful content and functionality, that label seems part of U-Haul's corporate-speak, not the language of users.

Products & Services - If I wanted a hand truck, I'd look under "Hand trucks," not "Dollies." This disconnect may be due to regional differences: U-Haul is based in Phoenix, and I'm from New York. But which is the more common usage? Or if both labels are comparably common, should U-Haul list both terms?

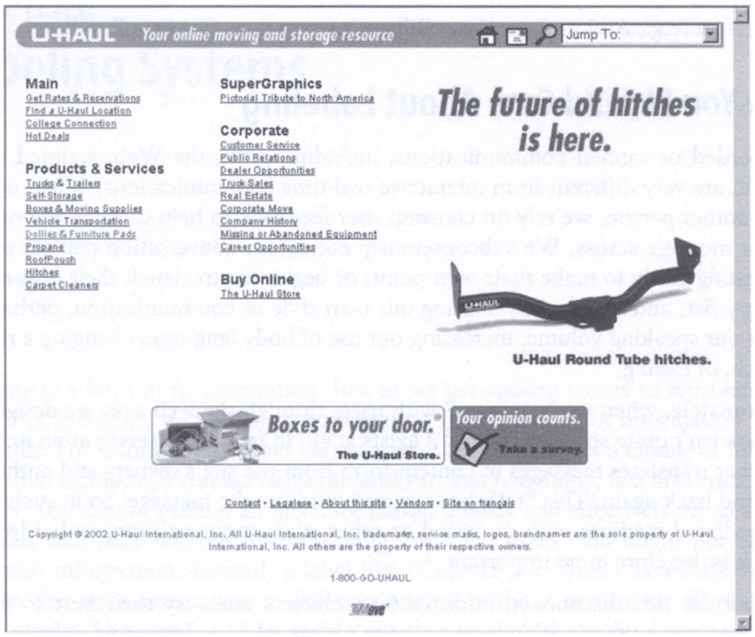
SuperGraphics - Have you ever heard this term before? They're not graphics, they're apparently something better ("super"). English is wonder-fully flexible, and new words are invented every day. But it's not realistic to expect impatient users to catch up with your linguistic creativity. Are "SuperGraphics" as important as "Products & Services"? What will we find behind the link "Pictorial Tribute to North American" photos, a travelogue? And just what does such a tribute have to do with leasing trucks anyway?

Corporate **-** Do users understand what "Corporate" means? The term sounds, well, rather corporate, as if it might be intended for employees, suppliers, and others involved with the corporation. Perhaps the more conventional label "About Us" might be more appropriate. "Corporate Move" is a service for corporate relocations, not anything about U-Haul moving to the new headquarters. Other links don't seem to belong here: like "Corporate Move," " Truck Sales" seems like it should go under "Products & Services." "Real Estate" and "Missing or Abandoned Equipment" are oddities that doesn't seem to belong anywhere. Is "Corporate" really another way of saying "Miscellaneous"

Buy Online **-** Like "SuperGraphics" this label describes a single link, which is wasteful. And that link, The U-Haul Store" seems to be a place to purchase or lease products and services. Why is "The U-Haul Store" not set aside here? Does U-Haul want to accentuate it for some reason? If that reason has little to do with users, perhaps it's got everything to do with internal politics – perhaps one of U-Haul VP owns "Products & Services," another owns "The U-Haul Store," and until they battle out their turf issues and one is extinguished, never the twain shall meet.

The results of this quick exercise can be split into these categories:

The labels aren't representative and don't differentiate **-** Too many of U-Haul's labels don't represent the content they link to or precede. Other than clicking through, users have no way to learn what "Corporate Move" means or what the difference is between "Products & Services" and "The U-Haul Store." Groupings of dissimilar items (e.g., "Trucks Sales," "Public Relations," and "Missing or Abandoned Equipment") don't provide any context for what those items' labels really represent. There is too much potential or confusion to consider these labels effective.



Main
Get Rates & Reservations
Find a U-Haul Location
College Connection
Hot Deals

Products & Services
Trucks & Trailers
Self-Storage
Boxes & Moving Supplies
Vehicle Transportation
Dollies & Furniture Pads
Propane
RoofPouch
Hitches
Carpet Cleaners

SuperGraphics
Pictorial Tribute to North America

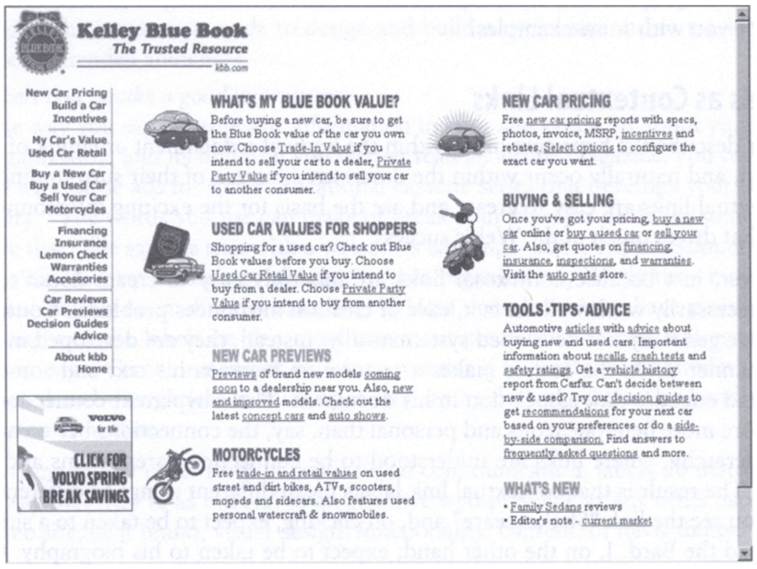
Corporate
Customer Service
Public Relations
Dealer Opportunities
Truck Sales
Real Estate
Corporate Move
Company History
Missing or Abandoned Equipment
Career Opportunities

Buy Online
The U-Haul Store

Contact · Legalese · About this site · Vendors · Site en français

Copyright © 2002 U-Haul International, Inc. All U-Haul International, Inc. trademarks, service marks, logos, brandnames are the sole property of U-Haul International, Inc. All others are the property of their respective owners.

1-800-GO-UHAUL



New Car Pricing
Build a Car
Incentives

My Car's Value
Used Car Retail

Buy a New Car
Buy a Used Car
Sell Your Car
Motorcycles

Financing
Insurance
Lemon Check
Warranties
Accessories

Car Reviews
Car Previews
Decision Guides
Advice

About kbb
Home

WHAT'S MY BLUE BOOK VALUE?
Before buying a new car, be sure to get the Blue Book value of the car you own now. Choose Trade-In Value if you intend to sell your car to a dealer, Private Party Value if you intend to sell your car to another consumer.

USED CAR VALUES FOR SHOPPERS
Shopping for a used car? Check out Blue Book values before you buy. Choose Used Car Retail Value if you intend to buy from a dealer. Choose Private Party Value if you intend to buy from another consumer.

NEW CAR PREVIEWS
Previews of brand new models coming soon to a dealership near you. Also, new and improved models. Check out the latest concept cars and auto shows.

MOTORCYCLES
Free trade-in and retail values on used street and dirt bikes, ATVs, scooters, mopeds and sidecars. Also features used personal watercraft & snowmobiles.

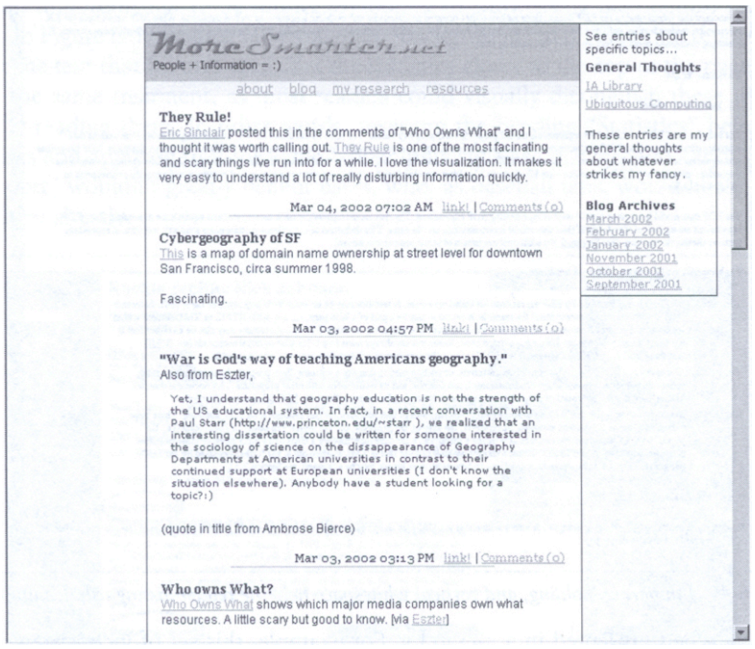
NEW CAR PRICING
Free new car pricing reports with specs, photos, invoice, MSRP, incentives and rebates. Select options to configure the exact car you want.

BUYING & SELLING
Once you've got your pricing, buy a new car online or buy a used car or sell your car. Also, get quotes on financing, insurance, inspections, and warranties. Visit the auto parts store.

TOOLS·TIPS·ADVICE
Automotive articles with advice about buying new and used cars. Important information about recalls, crash tests and safety ratings. Get a vehicle history report from Carfax. Can't decide between new & used? Try our decision guides to get recommendations for your next car based on your preferences or do a side-by-side comparison. Find answers to frequently asked questions and more.

WHAT'S NEW
• Family Sedans reviews
• Editor's note - current market

VOLVO for life
CLICK FOR VOLVO SPRING BREAK SAVINGS


MoreSmarter.net
People + Information = :)
about blog my research resources

They Rule!
Eric Sinclair posted this in the comments of "Who Owns What" and I thought it was worth calling out. They Rule is one of the most facinating and scary things I've run into for a while. I love the visualization. It makes it very easy to understand a lot of really disturbing information quickly.

Mar 04, 2002 07:02 AM link! | Comments (0)

Cybergeography of SF
This is a map of domain name ownership at street level for downtown San Francisco, circa summer 1998.

Fascinating.

Mar 03, 2002 04:57 PM link! | Comments (0)

"War is God's way of teaching Americans geography."
Also from Eszter,

> Yet, I understand that geography education is not the strength of the US educational system. In fact, in a recent conversation with Paul Starr (http://www.princeton.edu/~starr), we realized that an interesting dissertation could be written for someone interested in the sociology of science on the dissappearance of Geography Departments at American universities in contrast to their continued support at European universities (I don't know the situation elsewhere). Anybody have a student looking for a topic?:)

(quote in title from Ambrose Bierce)

Mar 03, 2002 03:13 PM link! | Comments (0)

Who owns What?
Who Owns What shows which major media companies own what resources. A little scary but good to know. [via Eszter]

See entries about specific topics...
General Thoughts
IA Library
Ubiquitous Computing

These entries are my general thoughts about whatever strikes my fancy.

Blog Archives
March 2002
February 2002
January 2002
November 2001
October 2001
September 2001

***The labels are jargony, not user-centric* -** Labels like “College Connection” and “SuperGraphics” can expose an organization that, despite its best intentions, does not consider the importance of its customers’ needs as important as its own goals, politics, and culture. This is often the case when the web sites use organizational jargon for their labels. You’ve probably seen such sites; their labels are crystal clear, obvious, and enlightening, as long you’re one of the .01 percent of users who actually work for the sponsoring organization. A sure way to lose a sale is to label your site’s product ordering system as an “Orde-ring Processing and Fulfillment Facility.”

***The labels waste money* -** There are too many chances for a user to step into one of the many confusing cognitive traps presented by U-Haul’s labels. And any time an architecture intrudes on a user’s experience and forces him to pause and sat “huh?”, there is a reasonable chance that he will give up on a site and go somewhere else, especially given the competitive nature of this medium. In other words, confusing labels can negate the investment made to design and build a useful site and to market that site to intended audiences.

***The labels don't make a good impression* -** The way you say or represent information in your site says a lot about you, your organization, and its brand. If you've ever read an airline magazine, you're familiar with those ads for some educational cassette series that develops your vocabulary. "The words you use can make or break your business deals" or something like that. The same is true with a web site's labeling-poor, unprofessional labeling can destroy a user's confidence in that organization. While they may have spent heavily on traditional branding, U-Haul doesn't seem to have given much thought to the labels on this most important piece of its virtual real estate, its main page. Customers might be left to wonder if U-Haul will be similarly hap-hazard and thoughtless in the way it services its fleet of vehicles or handles the customer hotline.

Like writing or any other form of professional communication, labels do matter. It's fair to say that they're as integral to an effective web presence as any other aspect of your web site, be it brand, visual design, functionality, content, or navigability.

Varieties of Labels
On the Web, we regularly encounter labels in two formats: textual and iconic. In this chapter, we'll spend most of our time addressing textual labels, as they remain the most common despite the Web's highly visual nature, including:

***Contextual Links* -** Hyperlinks to chunks of information on other pages or to another location on the same page.

***Headings* -** Labels that simply describe the content that follows them, just as print headings do.

***Navigation System Choices* -** Labels representing the options in navigation systems.

***Index Terms* -** Keywords and subject headings that represent content for searching or browsing.

These categories are by no means perfect or mutually exclusive. A single label can do double duty; for example, the contextual link "Naked Bungee Jumping" could lead to a page that uses the heading label "Naked Bungee Jumping" and has been indexed as being about (you guessed it) "Naked Bungee Jumping." And some of these labels could be iconic rather than textual, although we'd rather not imagine a visual representation of naked bungee jumping. In the following section, we'll explore the varieties of labeling in greater detail and provide you with some examples.

Labels as Contextual Links
Labels describe the hypertext links within the body of a document or chunk of information, and naturally occur within the descriptive context of their surrounding text. Contextual links are easy to create and are the basis for the exciting intercon-nectedness that drives much of the Web's success. However, just because contextual links are relatively easy to create doesn't mean they necessarily work well. In fact, ease of creation introduces problems. Contextual links are generally not developed systematically; instead, they are developed in an ad hoc manner when the author makes a connection between his text and something else, and encodes that association in his document. These hypertext connections are therefore more heterogeneous and personal than, say, the connections between items in a hierarchy, where links are understood to be connecting parent items and child items. The result is that contextual link labels mean different things to different people. You see the link "Shakespeare" and, on clicking, expect to be taken to a site dedicated to the Bard. I, on the other hand, expect to be taken to his biography. Or an image. Or a Usenet newsgroup.

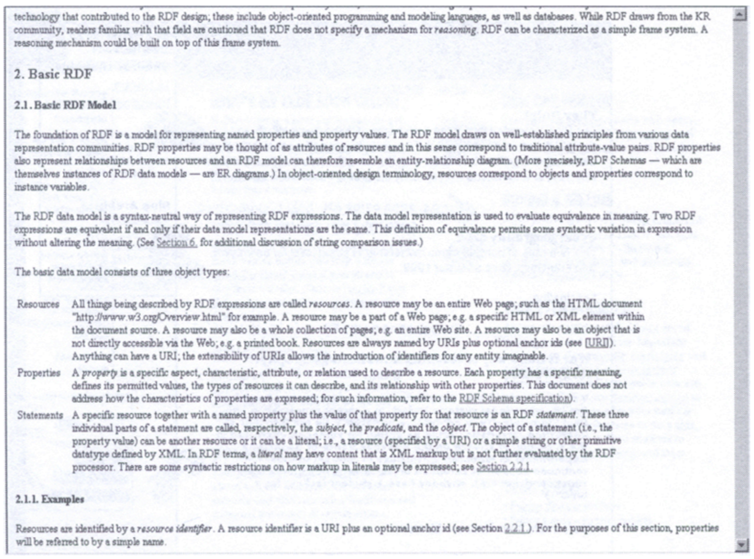
To be more representational of the content they connect to, contextual links rely instead upon, naturally, context. If the content's author suc-ceeds at establishing that context in his writing, then the label draws meaning from its surrounding text. If he doesn't, the label loses its representational value.

Because Kelley Blue Book is a site dedicated to providing information to car buyers and sellers, contextual links need to be straightforward and meaningful. Kelley's contextual link labels, such as "Trade-In Value" and "Sell Your Car," are representational, and draw on surrounding text and headings to make it clear what type of help you'll receive if you click through. These highly representational labels are made even clearer by their context: explanatory text, clear headings, and a site that itself has a few straightforward uses.

On the other hand, contextual links on a personal web log (“blog") aren't necessarily so clear. The author is among friends, and can assume that his loyal regular readers possess a certain level of background-really, contextual knowledge in their heads. Or he knows that keeping his link labels less representational creates some mystery around what they will lead to. So the author may choose to design contextual link labels that aren't so representational. E.g. the author expects us to know who "Eric Sinclair" is perhaps he's been mentioned in this blog before. Or the author knows that we'll recognize the label "Eric Sinclair" as a person, and provides some minimal Context-the fact that Eric wrote some comments-to entice the user to click through. "They Rule" is equally mysterious; we have no idea what this label represents, but the blog author contextualizes it as "fascinating" and "scary." Nonrepre-sentational labels have their place; as it's likely that we already trust the blog author's opinion, we'll probably want to click through and learn more. But if that trust doesn't always exist, nonrepresentational links can be damaging.

As we'll see, other varieties of labels derive context, and therefore meaning, from being pan of larger sets of labels or labeling systems. But systematic consistency isn't quite so possible with link labels. These labels are instead glued together by the copy and context. However, consistency between these labels and the chunks of information they link to does become an issue.

An information architect can ensure that contextual link labels are representational by asking, "What kind of information will the user expect to be taken to?" before creating and labeling a contextual link. Contextual links are created in such an ad hoc manner that simply asking this question will improve the quality of representation. (An easy way to study users' interpretations of labels is to provide a printout of a page with the labels clearly identified, and have subjects jot down what they'd expect each to link to.)



technology that contributed to the RDF design; these include object-oriented programming and modeling languages, as well as databases. While RDF draws from the KR community, readers familiar with that field are cautioned that RDF does not specify a mechanism for *reasoning*. RDF can be characterized as a simple frame system. A reasoning mechanism could be built on top of this frame system.

2. Basic RDF

2.1. Basic RDF Model

The foundation of RDF is a model for representing named properties and property values. The RDF model draws on well-established principles from various data representation communities. RDF properties may be thought of as attributes of resources and in this sense correspond to traditional attribute-value pairs. RDF properties also represent relationships between resources and an RDF model can therefore resemble an entity-relationship diagram. (More precisely, RDF Schemas — which are themselves instances of RDF data models — are ER diagrams.) In object-oriented design terminology, resources correspond to objects and properties correspond to instance variables.

The RDF data model is a syntax-neutral way of representing RDF expressions. The data model representation is used to evaluate equivalence in meaning. Two RDF expressions are equivalent if and only if their data model representations are the same. This definition of equivalence permits some syntactic variation in expression without altering the meaning. (See Section 6. for additional discussion of string comparison issues.)

The basic data model consists of three object types:

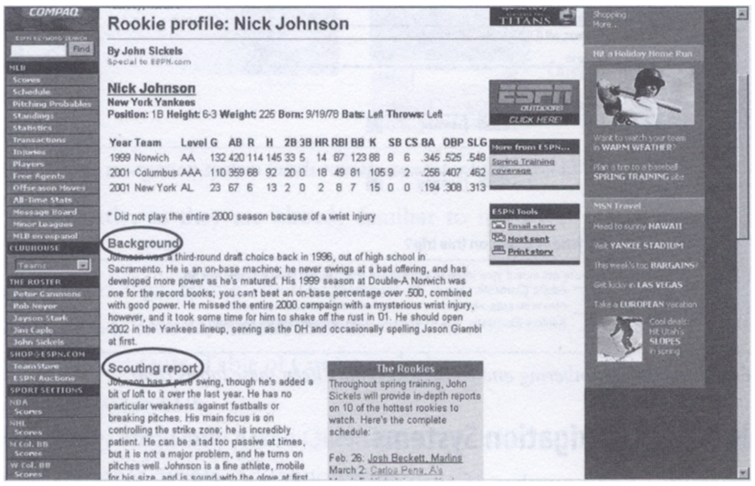
Resources	All things being described by RDF expressions are called *resources*. A resource may be an entire Web page; such as the HTML document "http://www.w3.org/Overview.html" for example. A resource may be a part of a Web page; e.g. a specific HTML or XML element within the document source. A resource may also be a whole collection of pages; e.g. an entire Web site. A resource may also be an object that is not directly accessible via the Web; e.g. a printed book. Resources are always named by URIs plus optional anchor ids (see [URI]). Anything can have a URI; the extensibility of URIs allows the introduction of identifiers for any entity imaginable.
Properties	A *property* is a specific aspect, characteristic, attribute, or relation used to describe a resource. Each property has a specific meaning, defines its permitted values, the types of resources it can describe, and its relationship with other properties. This document does not address how the characteristics of properties are expressed; for such information, refer to the RDF Schema specification).
Statements	A specific resource together with a named property plus the value of that property for that resource is an RDF *statement*. These three individual parts of a statement are called, respectively, the *subject*, the *predicate*, and the *object*. The object of a statement (i.e., the property value) can be another resource or it can be a literal; i.e., a resource (specified by a URI) or a simple string or other primitive datatype defined by XML. In RDF terms, a *literal* may have content that is XML markup but is not further evaluated by the RDF processor. There are some syntactic restrictions on how markup in literals may be expressed; see Section 2.2.1.

2.1.1. Examples

Resources are identified by a *resource identifier*. A resource identifier is a URI plus an optional anchor id (see Section 2.2.1.). For the purposes of this section, properties will be referred to by a simple name.

On the other hand, it's important to acknowledge that contextual links are often not within the information architect's control. Usually, content authors are responsible for contextual links. They are the ones who know the meaning of their content and how to best link it to other content. So while you may want to enforce rules for contextual link labels (such as what an employee's name should always link to), you may be better off suggesting guidelines to content authors (such as suggesting that employees' names link to a corresponding directory listing when possible).

Labels as Headings
Labels ate often used as headings that describe the chunk of information that follows. Headings, as shown in Figure 6-4, are often used to establish a hierarchy within a text. Just as in a book, where headings help us distinguish chapters from sections, they also help us determine a site's subsites, or differentiate categories from subcategories.



Rookie profile: Nick Johnson

By John Sickels
Special to ESPN.com

Nick Johnson
New York Yankees
Position: 1B **Height:** 6-3 **Weight:** 225 **Born:** 9/19/78 **Bats:** Left **Throws:** Left

Year	Team	Level	G	AB	R	H	2B	3B	HR	RBI	BB	K	SB	CS	BA	OBP	SLG
1999	Norwich	AA	132	420	114	145	33	5	14	87	123	88	8	6	.345	.525	.548
2001	Columbus	AAA	110	359	68	92	20	0	18	49	81	105	9	2	.256	.407	.462
2001	New York	AL	23	67	6	13	2	0	2	8	7	15	0	0	.194	.308	.313

* Did not play the entire 2000 season because of a wrist injury

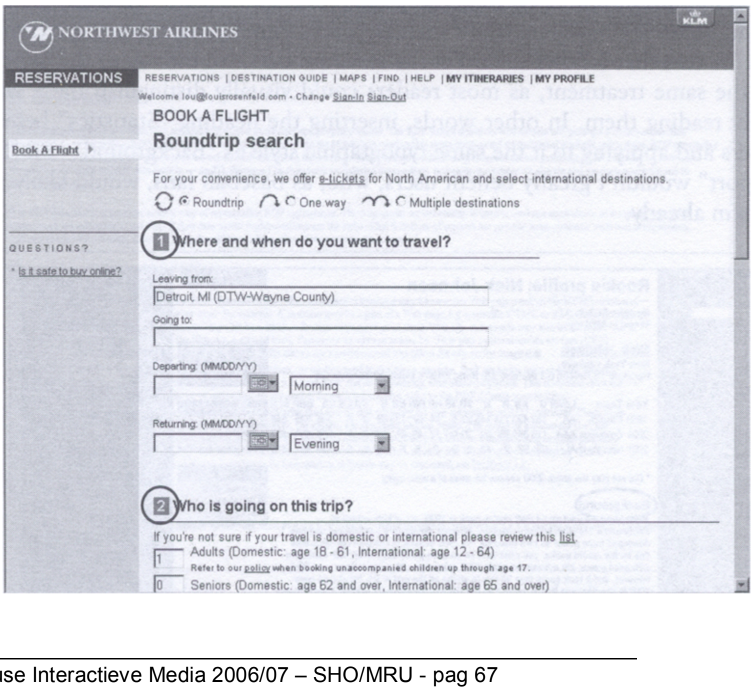
Background
Johnson was a third-round draft choice back in 1996, out of high school in Sacramento. He is an on-base machine; he never swings at a bad offering, and has developed more power as he's matured. His 1999 season at Double-A Norwich was one for the record books; you can't beat an on-base percentage *over* .500, combined with good power. He missed the entire 2000 campaign with a mysterious wrist injury, however, and it took some time for him to shake off the rust in '01. He should open 2002 in the Yankees lineup, serving as the DH and occasionally spelling Jason Giambi at first.

Scouting report
Johnson has a pure swing, though he's added a bit of loft to it over the last year. He has no particular weakness against fastballs or breaking pitches. His main focus is on controlling the strike zone; he is incredibly patient. He can be a tad too passive at times, but it is not a major problem, and he turns on pitches well. Johnson is a fine athlete, mobile for his size, and is sound with the glove at first

The Rookies
Throughout spring training, John Sickels will provide in-depth reports on 10 of the hottest rookies to watch. Here's the complete schedule:

Feb. 26: Josh Beckett, Marlins
March 2: Carlos Pena, A's

The hierarchical relationships between headings-whether patent, child, or sibling are usually established visually through consistent use of numbering, font sizes, colors and styles, white space and indentation, or combinations thereof. A visually clear hierarchy, often the work of information or graphic designers, can take some pressure off information architects by reducing the need to create labels that convey that hierarchy. So a set of labels that don't mean much can suddenly take on meaning when presented in a hierarchy. For example, this set of


NORTHWEST AIRLINES
KLM
RESERVATIONS
RESERVATIONS | DESTINATION GUIDE | MAPS | FIND | HELP | MY ITINERARIES | MY PROFILE
Book A Flight
QUESTIONS?
* Is it safe to buy online?

Welcome lou@louisrosenfeld.com • Change Sign-In Sign-Out

BOOK A FLIGHT
Roundtrip search

For your convenience, we offer e-tickets for North American and select international destinations.

[✓ Roundtrip] One way Multiple destinations

1 Where and when do you want to travel?

Leaving from: Detroit, MI (DTW-Wayne County)
Going to:
Departing: (MM/DD/YY) Morning
Returning: (MM/DD/YY) Evening

2 Who is going on this trip?

If you're not sure if your travel is domestic or international please review this list.
1 Adults (Domestic: age 18 - 61, International: age 12 - 64)
Refer to our policy when booking unaccompanied children up through age 17.
0 Seniors (Domestic: age 62 and over, International: age 65 and over)

inconsistent headings may be quite confusing:

- Our Furniture Selection
- Office Chairs
- Our buyer's picks
- Chairs from Steelcase
- Hon products
- Herman Miller
- Aerons
- Lateral Files

However, they are much more meaningful when presented in a hierarchy:

- Our Furniture Selection
- Office Chairs
- Our buyer's picks
- Chairs from Steelcase
- Hon products
- Herman Miller
- Aerons
- Lateral Files

It's also important not to be too rigidly bound to showcasing hierarchical relationships. The heading labels such as "Background" and "Scouting report" represent the text that follows each. Yet the statistics closer to the top of the page don't merit the same treatment, as most readers could visually distinguish these without actually reading them. In other words, inserting the heading "Statistics" before the numbers and applying to it the same typographic style as "Background" and "Scouting report" wouldn't greatly benefit users, who, as baseball fans, would likely recognize them already. It is interesting to note, however, that it's difficult to distinguish one column of statistics from another, so each utilizes its own heading label.

We can be a bit more flexible when designing hierarchical headings, but it's especially important to maintain consistency when labeling steps in a process. To successfully navigate a process, it's typically necessary for users to complete each step along the way, so heading labels have to be obvious, and must also convey sequence. Using numbers is common, and consistently framing the labels as actions-utilizing verbs-also helps tie together the sequence of steps. In effect, the labels need to tell users where to start, where to go next, and what action will be involved in each step along the way. On the page from Northwest Airlines the heading labels are clearly numbered, are consistently laid out, and utilize a consistent syntax that describes the question addressed in each step of the process.

Heading labels, whether hierarchical or sequenced, come in multiples, and should be more systematically designed than contextual link labels.

Labels Within Navigation Systems
With typically a small number of options, navigation system labels demand consistent application more than any other type of label. A single inconsistent option can introduce an "apples and oranges" effect more quickly to a navigation system, which usually has fewer than ten choices, than to a set of index terms, which might have thousands. Additionally, a navigation system typically occurs again and again throughout a site, so navigation labeling problems are magnified through repeated exposure.

Users rely on a navigation system to behave "rationally" through consistent page location and look; labels should be no different. Effectively applied labels are integral to building a sense of familiarity, so they'd better not change from page to page. That's why using the label "Main" on one page, "Main Page" on another, and "Home" elsewhere could destroy the familiarity that the user needs when navigating a site. In the left-hand navigation system's three labels "Shop," "Learn," and "Travel" are applied consistently throughout the site, and would be even more effective if colors and locations were also consistent.

There are no standards, but some common variants exist for many navigation system labels. You should consider selecting one from each these categories and applying them consistently, as they are already familiar to most web users. Here is a non-exhaustive list:

- Main, Main Page, Home
- Search, Find, Browse, Search/Browse
- Site Map, Contents, Table of Contents, Index
- Contact, Contact Us
- Help, FAQ, Frequently Asked Questions
- News, News & Events, Announcements
- About, About Us, About <company name>, Who We Are

Of course, the same label can often represent different kinds of information. For example, in one site "News" may link to an area that includes announcements of new additions to the site. In another site "News" may link to an area of news stories describing national and world events. Obviously, if you use the same labels in different ways within your own site, your users will be very confused.

To address both problems, navigational labels can be augmented by brief descriptions (also known as scope notes) when initially introduced on the main page. The navigation system labels appear in brief on the left-hand side, and are described with scope notes in the body of the main page.

In this case, if more representational navigation system labels had been used in the first place, they may have diminished the need to devote so much valuable main page real estate to scope notes. There are alternatives to scope notes that don't monopolize real estate, such as JavaScript rollovers and other scripted mouse over effects, but these aren't conventional and don't work with all browsers. If you feel that your site will be regularly used by a loyal set of users who are willing to learn your site's conventions, then it's worth considering these alternatives; otherwise, we suggest sticking with web-wide conventions.

Labels as Index Terms
Often referred to as keywords, descriptive metadata, taxonomies, controlled vocabularies, and thesauri, sets of index term labels can be used to describe any type of content: sites, subsites, pages, content components, and so on. By representing the meaning of a piece of content, index terms support more precise searching than simply searching the full text of content someone has assessed the content's meaning and described it using index terms, and searching those terms ought to be more effective than having a search engine match a query against the content's full text.

Index terms are also used to make browsing easier: the meta data from a collection of documents can serve as the source of browsable lists or menus. This can be highly beneficial to users, as index terms provide an alternative to a site's primary organization system, such as an information architecture organized by business unit. Index terms in the form of site indexes and other lists provide a valuable alternative view by "cutting across the grain" of organizational silos.

In the index of the Sun Microsystems site is generated from index term labels, which, in turn, are used to identify content from many different Sun business units. Much of the content already accessible through Sun's primary organization system is also accessible by browsing these index terms (e.g., keywords).

Frequently, index terms are completely invisible to users. The records we use to rep- resent documents in content management systems and other databases typically include fields for index terms, which are often heard but not seen: they come into play only when you search. Similarly, index terms may be hidden as embedded meta- data in an HTML document's <META...> or <TITLE> tags. For example, a furniture manufacturer's site might list the following index terms in the <META...> tags of records for its upholstered items:

So a search on “sofa" would retrieve the page with these index terms even if the term “sofa" doesn't appear anywhere in the page's text. A search for “snack" retrieves this recipe, though there is no mention of the term in the recipe itself. “Snack" is likely stored separately as an index term in a database record for this recipe.

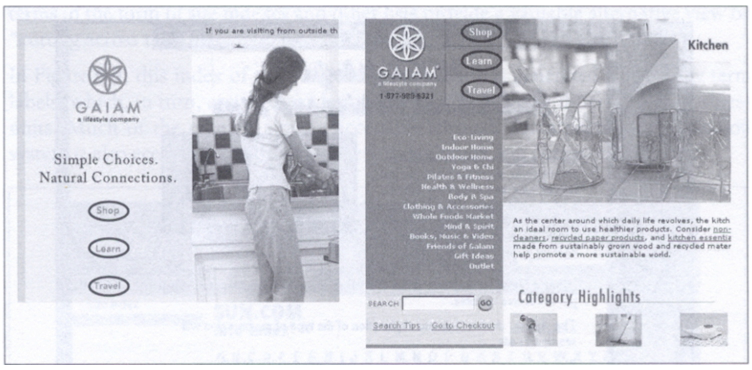
It's interesting how many sites' main pages don't feature index terms. Organizations do crazy, expensive things to get their sites noticed, including advertising their URL on banners flown over football stadiums. But using index terms to describe a main page is a much cheaper way for getting that page, and the site as a whole, indexed and "known" so that users who search the Web ate mote likely to find it.

Getting your pages to stand out from each other is a different and much mote daunting challenge. That's where a mote systematic approach to labeling – using index terms from controlled vocabularies or thesauri – has mote value. These sets of labels are designed to describe a delineated domain-such as products and services, or oncology- and to do so in a consistent, predictable manner.

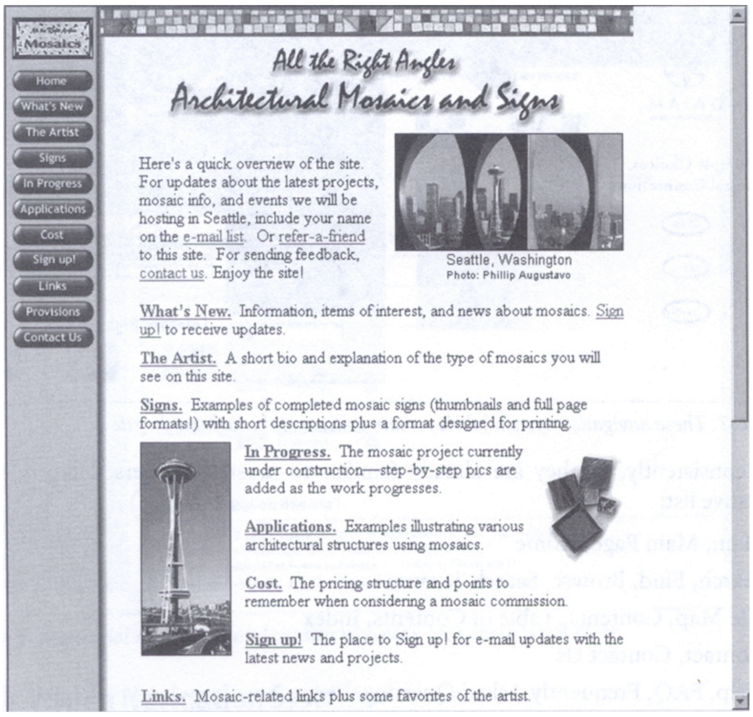
Iconic Labels
It's true that a picture is worth a thousand words. But which thousand?

Icons can represent information in much the same way as text can. We see them most frequently used as navigation system labels. Additionally, icons occasionally serve as heading labels and have even been known to show up as link labels, although this is rare.

The problem with iconic labels is that they constitute a much more limited language than text. That's why they're



As the center around which daily life revolves, the kitch an ideal room to use healthier products. Consider non- cleaners, recycled paper products, and kitchen essentia made from sustainably grown wood and recycled mater help promote a more sustainable world.



All the Right Angles
Architectural Mosaics and Signs

Here's a quick overview of the site. For updates about the latest projects, mosaic info, and events we will be hosting in Seattle, include your name on the e-mail list. Or refer-a-friend to this site. For sending feedback, contact us. Enjoy the site!

Seattle, Washington
Photo: Phillip Augustavo

What's New. Information, items of interest, and news about mosaics. Sign up! to receive updates.

The Artist. A short bio and explanation of the type of mosaics you will see on this site.

Signs. Examples of completed mosaic signs (thumbnails and full page formats!) with short descriptions plus a format designed for printing.

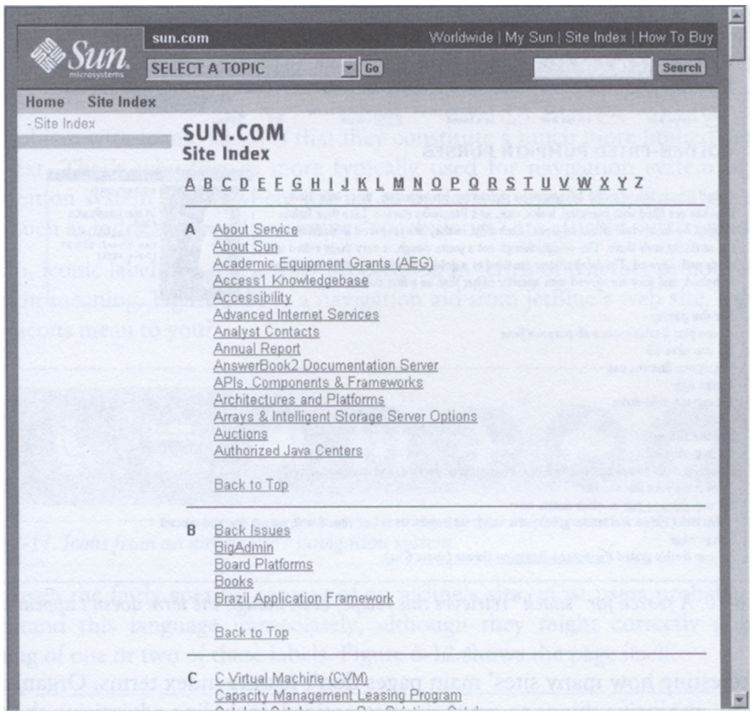
In Progress. The mosaic project currently under construction—step-by-step pics are added as the work progresses.

Applications. Examples illustrating various architectural structures using mosaics.

Cost. The pricing structure and points to remember when considering a mosaic commission.

Sign up! The place to Sign up! for e-mail updates with the latest news and projects.

Links. Mosaic-related links plus some favorites of the artist.



SUN.COM
Site Index

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

A About Service
About Sun
Academic Equipment Grants (AEG)
Access1 Knowledgebase
Accessibility
Advanced Internet Services
Analyst Contacts
Annual Report
AnswerBook2 Documentation Server
APIs, Components & Frameworks
Architectures and Platforms
Arrays & Intelligent Storage Server Options
Auctions
Authorized Java Centers

Back to Top

B Back Issues
BigAdmin
Board Platforms
Books
Brazil Application Framework

Back to Top

C C Virtual Machine (CVM)
Capacity Management Leasing Program

more typically used for navigation system or small organization system labels, where the list of options is small, than for larger sets of labels such as index terms, where iconic "vocabularies" are quickly outstripped.

Even so, iconic labels are still a risky proposition in terms of whether or not they can represent meaning. JetBlue's web site uses a iconic navigation aid. But what do the icons mean to you?

Even given the fairly specific context of an airline's site, most users probably won't understand this language immediately, although they might correctly guess the meaning of one or two of these labels.

Since the iconic labels are presented with textual labels, our test wasn't really fair. But it is interesting to note that even the site's designers acknowledge that the iconic labels don't stand well on their own and hence need textual explanations.

Iconic labels like these add aesthetic quality to a site, and as long as they don't compromise the site's usability, there's no reason not to use them. In fact, if your site's users visit regularly, the iconic "language" might get established in their minds through repeated exposure. In such situations, icons are an especially useful shorthand, both representational and easy to visually recognize – a double bonus. But it's interesting to note that jetBlue's subsidiary pages don't use iconic labels alone; they've chosen to maintain the icon/text pairing throughout their site. Unless your site has a patient, loyal audience of users who are willing to learn your visual language, we suggest using iconic labels only for systems with a limited set of options, being careful not to place form ahead of function.

Designing Labels
Designing effective labels is perhaps the most difficult aspect of information architecture. Language is simply too ambiguous for you to ever feel confident that you've perfected a label. There are always synonyms and homonyms to worry about, and different contexts influence our understanding of what a particular term means. But even labeling conventions are questionable: you absolutely cannot assume that the label "main page" will be correctly interpreted by 100.", of your site's users. Your labels will never be perfect, and you can only hope that your efforts make a difference, as measuring label effectiveness is extremely difficult if not impossible.

If it sounds to you like labeling is an art rather than a science, then you're absolutely correct. And as in all such cases, you can forget about finding incontrovertible rules, and hope for guidelines instead. Following are some guidelines and related issues that will help you as you delve into the mysterious an of label design.

General Guidelines
Remember that content, users, and context affect all aspects of an information architecture, and this is particularly true with labels. Any of the variables attached to users, content, and context can drag a label into the land of ambiguity.

Let's go back to the term "pitch." From baseball (what's thrown) to football (the field where it's played in the United Kingdom), from sales (what's sometimes made on the golf course) to sailing (the angle of the boat in the water), there are at least 15 different definitions, and it's hard to make sure that your site's users, content, and context will converge upon the same definition. This ambiguity makes it difficult to assign labels to describe content, and difficult for users to rely upon their assumptions about what specific labels actually mean.

So what can we do to make sure our labels are less ambiguous and more representational? The following two guidelines may help.

Narrow scope whenever possible - If we focus our sites on a more defined audience, we reduce the number of possible perspectives on what a label means. Sticking to fewer subject domains achieves more obvious and effective representation. A narrower business context means clearer goals for the site, its architecture, and therefore its labels.

To put it another way, labeling is easier if your site's content, users, and context are kept simple and focused. Too many sites have tried to take on too much, achieving broad mediocrity rather than nailing a few choice tasks. Accordingly, labeling systems often cover too much ground to truly be effective. If you are planning any aspect of your site's scope – who will use it, what content it will contain, and how, when, and why it should be used – erring toward simplicity will make your labels more effective.

If your site must be a jack of all trades, avoid using labels that address the entire site's content. (The obvious exception are the labels for site-wide navigation systems, which do cover the entire site.) But in the other areas of labeling, modularizing and simplifying content into subsites that meet the needs of specific audiences will enable you to design more modular, simpler collections of labels to address those specific areas.

This modular approach may result in separate labeling systems for different areas of your site. For example, records in your staff directory might benefit from a specialized labeling system that wouldn't make sense for other parts of the site, while your site-wide navigation system's labels wouldn't really apply to entries in the staff directory.

Develop consistent labeling systems, not labels **-** It's also important to remember that labels, like organization and navigation systems, are systems in their own right. Some are planned systems, some aren't. A successful system is designed with one or more characteristics that unify its members. In successful labeling systems, one characteristic is typically consistency.

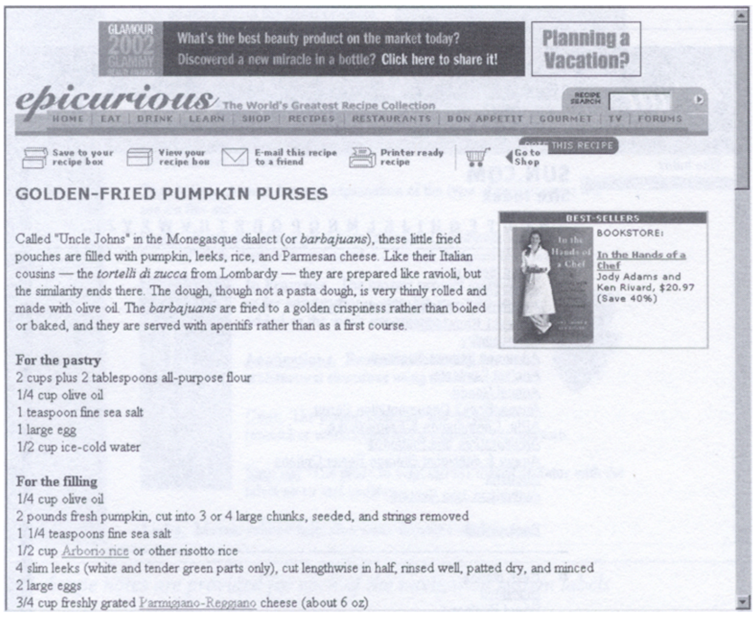
Why is consistency important? Because consistency means predictability, and systems that are predictable are simply easier to learn. You see one or two labels, and then you know what to expect from the rest-if the system is consistent. This is especially important for first-time visitors to a site, but consistency benefits all users by making labeling easy to learn, easy to use, and therefore invisible.
Consistency is affected by many issues:

- *Style:* Haphazard usage of punctuation and case is a common problem within labeling systems, and can be addressed, if not eliminated, by using style guides. Consider hiring a proofreader and purchasing a copy of Strunk & White.
- *Presentation:* Similarly, consistent application of fonts, font sizes, colors, white space, and grouping can help visually reinforce the systematic nature of a group of labels.
- *Syntax:* It's not uncommon to find verb-based labels (e.g., "Grooming Your Dog"), noun-based labels (e.g., "Diets for Dogs"), and question-based labels (e.g., "How Do You Paper-Train Your Dog?") all mixed together. Within a

specific labeling system, consider choosing a single syntactical approach and sticking with it.

– *Granularity*: Within a labeling system, it can be helpful to present labels that are roughly equal in their specificity. Exceptions (such as site indexes) aside, it can be confusing to encounter a set of labels that cover differing levels of granularity. Here is an example: "Chinese restaurants," "Restaurants," "Taquerias," "Fast Food Franchises," "Burger Kings."
– *Comprehensiveness:* Users can be tripped up by noticeable gaps in a labeling system. For example, if a clothing retailer's site lists "pants," "ties," and "shoes," while somehow omitting "shirts," we may feel like something's wrong. Do they really not carry shirts? Or did they make a mistake? Aside from improving consistency, a comprehensive scope also helps users do a better job of quickly scanning and inferring the content a site will provide.
– *Audience:* Mixing terms like "lymphoma" and "tummy-ache" in a single labeling system can also throw off users, even if only temporarily. Consider the languages of your site's major audiences. If each audience uses a very different terminology, you may have to develop a separate labeling system for each audience, even if these systems are describing exactly the same content.

There are other potential roadblocks to consistency. None is particularly difficult to address, but you can certainly save a lot of labor and heartache if you consider these issues before you dive into creating labeling systems.

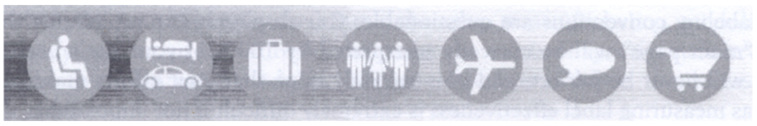

GLAMOUR 2002 GLAMMY
What's the best beauty product on the market today?
Discovered a new miracle in a bottle? Click here to share it!
Planning a Vacation?
epicurious The World's Greatest Recipe Collection
RECIPE SEARCH
HOME | EAT | DRINK | LEARN | SHOP | RECIPES | RESTAURANTS | BON APPETIT | GOURMET | TV | FORUMS
Save to your recipe box
View your recipe box
E-mail this recipe to a friend
Printer ready recipe
Go to Shop
THIS RECIPE
BEST-SELLERS
BOOKSTORE:
In the Hands of a Chef
Jody Adams and Ken Rivard, $20.97 (Save 40%)

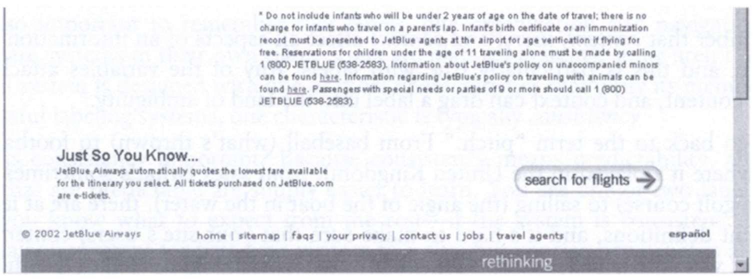
GOLDEN-FRIED PUMPKIN PURSES

Called "Uncle Johns" in the Monegasque dialect (or *barbajuans*), these little fried pouches are filled with pumpkin, leeks, rice, and Parmesan cheese. Like their Italian cousins — the *tortelli di zucca* from Lombardy — they are prepared like ravioli, but the similarity ends there. The dough, though not a pasta dough, is very thinly rolled and made with olive oil. The *barbajuans* are fried to a golden crispiness rather than boiled or baked, and they are served with aperitifs rather than as a first course.

For the pastry
2 cups plus 2 tablespoons all-purpose flour
1/4 cup olive oil
1 teaspoon fine sea salt
1 large egg
1/2 cup ice-cold water

For the filling
1/4 cup olive oil
2 pounds fresh pumpkin, cut into 3 or 4 large chunks, seeded, and strings removed
1 1/4 teaspoons fine sea salt
1/2 cup Arborio rice or other risotto rice
4 slim leeks (white and tender green parts only), cut lengthwise in half, rinsed well, patted dry, and minced
2 large eggs
3/4 cup freshly grated Parmigiano-Reggiano cheese (about 6 oz)

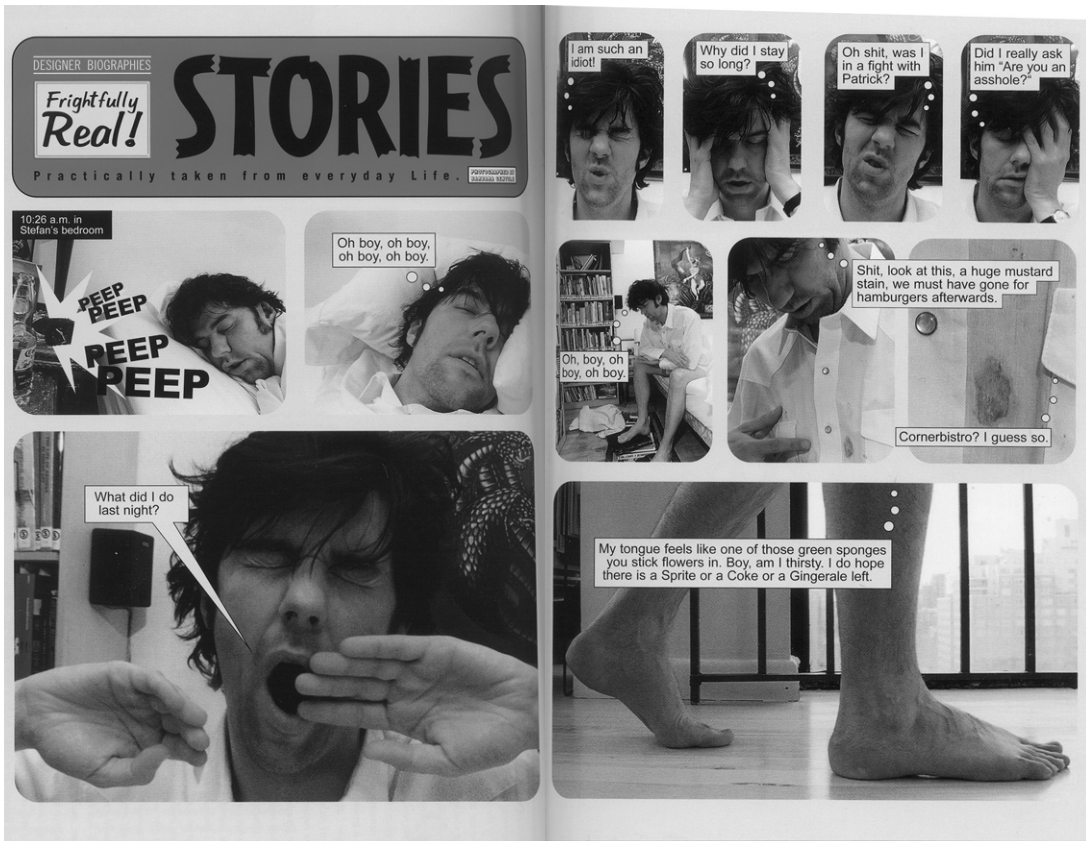




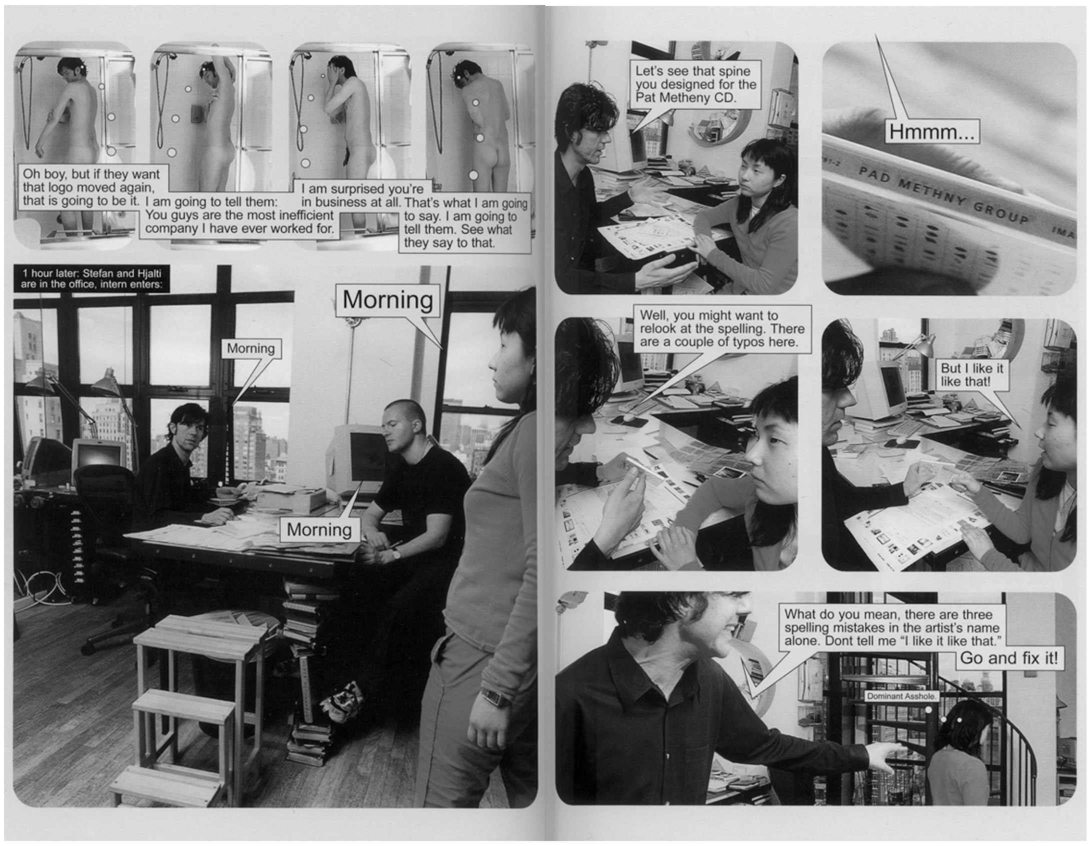
* Do not include infants who will be under 2 years of age on the date of travel; there is no charge for infants who travel on a parent's lap. Infant's birth certificate or an immunization record must be presented to JetBlue agents at the airport for age verification if flying for free. Reservations for children under the age of 11 traveling alone must be made by calling 1 (800) JETBLUE (538-2583). Information about JetBlue's policy on unaccompanied minors can be found here. Information regarding JetBlue's policy on traveling with animals can be found here. Passengers with special needs or parties of 9 or more should call 1 (800) JETBLUE (538-2583).

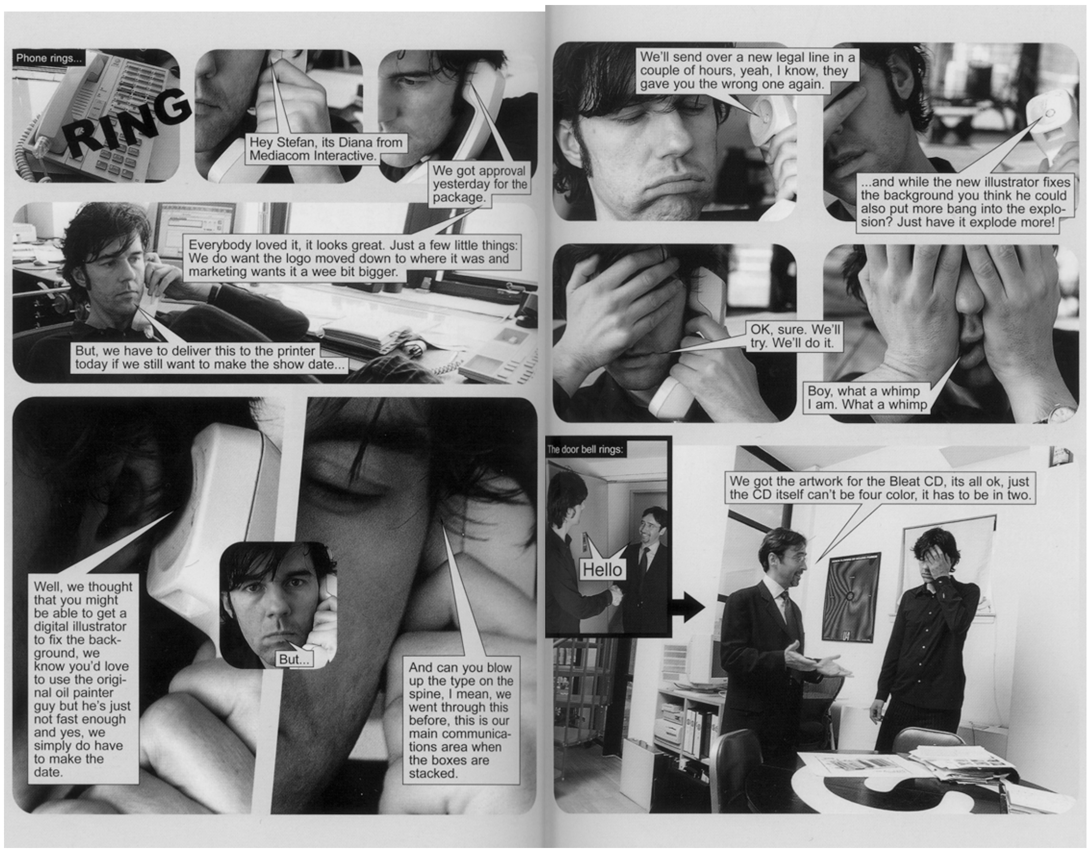
Just So You Know...
JetBlue Airways automatically quotes the lowest fare available for the itinerary you select. All tickets purchased on JetBlue.com are e-tickets.

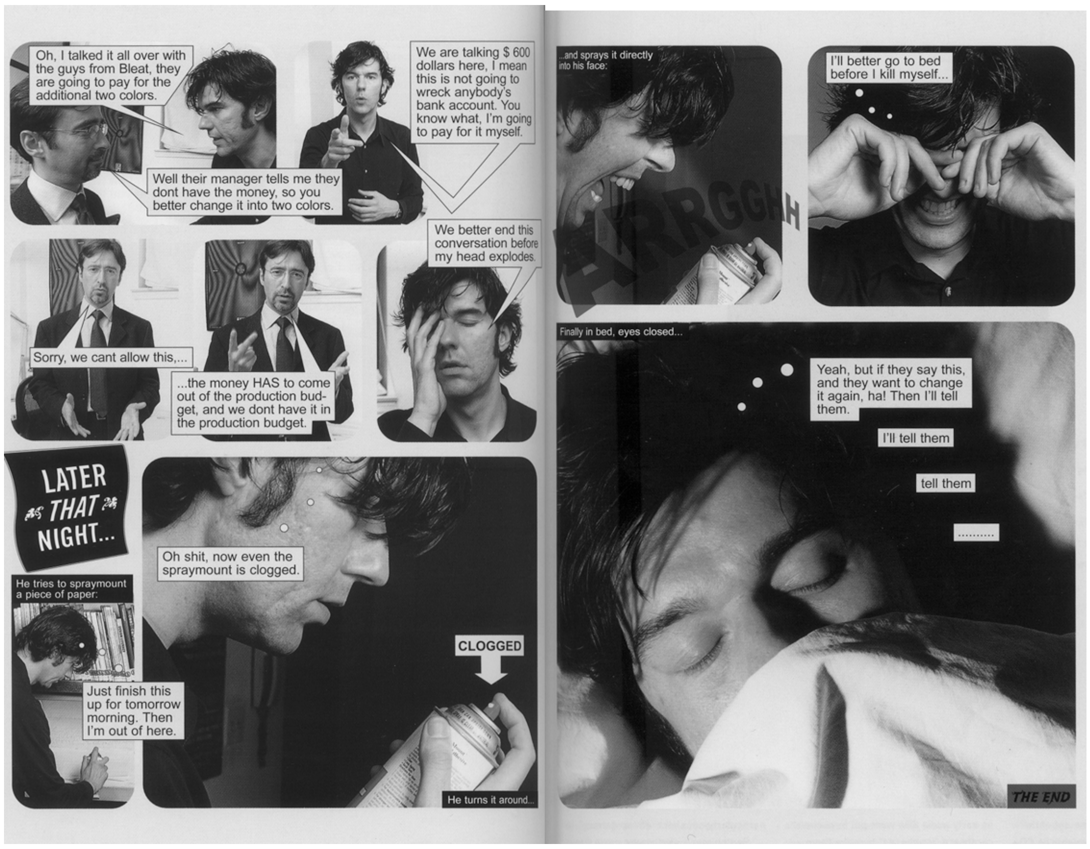
search for flights
© 2002 JetBlue Airways
home | sitemap | faqs | your privacy | contact us | jobs | travel agents
español
rethinking


DESIGNER BIOGRAPHIES
Frightfully Real!
STORIES
Practically taken from everyday Life.
10:26 a.m. in Stefan's bedroom
PEEP PEEP
PEEP PEEP
Oh boy, oh boy, oh boy, oh boy.
What did I do last night?
I am such an idiot!
Why did I stay so long?
Oh shit, was I in a fight with Patrick?
Did I really ask him "Are you an asshole?"
Oh, boy, oh boy, oh boy.
Shit, look at this, a huge mustard stain, we must have gone for hamburgers afterwards.
Cornerbistro? I guess so.
My tongue feels like one of those green sponges you stick flowers in. Boy, am I thirsty. I do hope there is a Sprite or a Coke or a Gingerale left.

Stefan opens fridge,
ouiiiiiiik
opens coke,
zschHHH
takes a long sip...
sip!
What the fuck? Shit, what is this? Soft and squashy?
and looks down:
AAARRRRRRGGHHH
UNBEFUCKING LIEVABLE !!!
FLASH BACK
One week earlier:
Whats going on? Mice? Rats?
Better put some rat poison out.
A couple of days later Stefan found an enormous dead rat in the apartment. It must have been pregnant.
a minute later in the shower...
What an unbelievable start to the day.
Just unbelievable....
...and we have to get that silly game packaging out to the printer.


Oh boy, but if they want that logo moved again, that is going to be it.
I am going to tell them: You guys are the most inefficient company I have ever worked for.
I am surprised you're in business at all.
That's what I am going to say. I am going to tell them. See what they say to that.
1 hour later: Stefan and Hjalti are in the office, intern enters:
Morning
Morning
Morning
Let's see that spine you designed for the Pat Metheny CD.
Hmmm...
PAD METHNY GROUP
Well, you might want to relook at the spelling. There are a couple of typos here.
But I like it like that!
What do you mean, there are three spelling mistakes in the artist's name alone. Dont tell me "I like it like that."
Go and fix it!
Dominant Asshole.


Phone rings...
RING
Hey Stefan, its Diana from Mediacom Interactive.
We got approval yesterday for the package.
Everybody loved it, it looks great. Just a few little things: We do want the logo moved down to where it was and marketing wants it a wee bit bigger.
But, we have to deliver this to the printer today if we still want to make the show date...
Well, we thought that you might be able to get a digital illustrator to fix the back-ground, we know you'd love to use the origi-nal oil painter guy but he's just not fast enough and yes, we simply do have to make the date.
But...
And can you blow up the type on the spine, I mean, we went through this before, this is our main communica-tions area when the boxes are stacked.
We'll send over a new legal line in a couple of hours, yeah, I know, they gave you the wrong one again.
...and while the new illustrator fixes the background you think he could also put more bang into the explo-sion? Just have it explode more!
OK, sure. We'll try. We'll do it.
Boy, what a whimp I am. What a whimp
The door bell rings:
Hello
We got the artwork for the Bleat CD, its all ok, just the CD itself can't be four color, it has to be in two.


Oh, I talked it all over with the guys from Bleat, they are going to pay for the additional two colors.
Well their manager tells me they dont have the money, so you better change it into two colors.
We are talking $ 600 dollars here, I mean this is not going to wreck anybody's bank account. You know what, I'm going to pay for it myself.
Sorry, we cant allow this,...
...the money HAS to come out of the production budget, and we dont have it in the production budget.
We better end this conversation before my head explodes.
LATER THAT NIGHT...
He tries to spraymount a piece of paper:
Just finish this up for tomorrow morning. Then I'm out of here.
Oh shit, now even the spraymount is clogged.
CLOGGED
He turns it around...
...and sprays it directly into his face:
HRRGGHH
I'll better go to bed before I kill myself...
Finally in bed, eyes closed...
Yeah, but if they say this, and they want to change it again, ha! Then I'll tell them.
I'll tell them
tell them
....
THE END

Uit: Tagmeister van Stefan Tagmeister